SANDIA REPORT
SAND87 — 1912 • UC — 32
Unlimited Release
Printed March 1989

PRONTO 3D
A Three-Dimensional Transient Solid Dynamics Program

L. M. Taylor, D. P. Flanagan

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550
for the United States Department of Energy
under Contract DE-AC04-76DP00789

153p

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Printed in the United States of America
Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161

NTIS price codes
Printed copy: A08
Microfiche copy: A01

SAND87-1912
Unlimited Release
Printed March 1989

Distribution
UC-32

PRONTO 3D A Three-Dimensional Transient Solid Dynamics Program

L. M. Taylor and D. P. Flanagan
Applied Mechanics Division III
Sandia National Laboratories
Albuquerque, New Mexico 87185

Abstract

PRONTO 3D is a three-dimensional transient solid dynamics code for analyzing large deformations of highly nonlinear materials subjected to extremely high strain rates. This Lagrangian finite element program uses an explicit time integration operator to integrate the equations of motion. Eight-node uniform strain hexahedral elements are used in the finite element formulation. A number of new numerical algorithms which have been developed for the code are described in this report. An adaptive time step control algorithm is described which greatly improves stability as well as performance in plasticity problems. A robust hourglass control scheme which eliminates hourglass distortions without disturbing the finite element solution is included. All constitutive models in PRONTO are cast in an unrotated configuration defined using the rotation determined from the polar decomposition of the deformation gradient. An accurate incremental algorithm was developed to determine this rotation and is described in detail. A robust contact algorithm was developed which allows for the impact and interaction of deforming contact surfaces of quite general geometry. Numerical examples are presented to demonstrate the utility of these algorithms.

Contents

Figures

Tables

1. INTRODUCTION

PRONTO 3D is a finite element program for the analysis of the three-dimensional response of solid bodies subjected to transient dynamic loading. The program includes nonlinear constitutive models, and accurately analyzes large deformations which may lead to geometric nonlinearities. PRONTO is a powerful tool for analyzing a wide variety of problems, including classes of problems in impact dynamics, rock blasting, and accident analyses.

PRONTO 3D is a direct descendant of the PRONTO 2D [1] code and readers will recognize the similarity between this report and Reference [1]. We have tried whenever possible to keep the theory and algorithms the same in both codes. The only notable exception to this is that the contact algorithm for the three-dimensional code is by necessity quite different.

We developed a flexible, problem-oriented language for the input to PRONTO which allows the user to define a complex mechanics problem with a few concise commands. The user instructions are similar in PRONTO 2D and PRONTO 3D. Experience has shown that after a user has gained some experience with the code, reference to the user's instructions (Appendix A) is seldom needed. There are no references to node or element numbers in the problem definition. All boundary conditions are specified through the concept of node and element side sets which are defined using the GENESIS [2] mesh definition data base. The GENESIS data base is a subset of the EXODUS [3] finite element data base. PRONTO contains no mesh generation or post-processing capabilities; it relies on external mesh generators and external post-processors. All post-processing of the finite element results is accomplished by accessing the EXODUS data base that PRONTO writes during the analysis.

The development of PRONTO was motivated by the need for a code which could serve as a testbed for research into numerical algorithms and new constitutive models for nonlinear materials. Towards this goal, the code contains a well-documented and easy-to-use interface for implementing new constitutive models (Appendix C). Complete documentation of the code architecture and computer storage requirements is provided in Appendix B.

PRONTO is written in completely standard FORTRAN [4]. Any system dependent coding (such as the determination of the date or the memory management) is part of the SUPES [5] package. The only input/output units used by PRONTO are 5, 6, 9, 11, 30 and 32. Their use is described in Table 1.1.

Table 1.1. INPUT/OUTPUT Units

Unit	Use
5	Formatted input instructions for PRONTO 3D
6	Formatted output from PRONTO 3D
9	Unformatted GENESIS mesh file
11	Unformatted EXODUS post-processing file
30	Unformatted restart output file
32	Unformatted restart input file

2. GOVERNING EQUATIONS

In this chapter, we present the underlying continuum mechanics concepts which are necessary to follow the development of the numerical algorithms in the following chapters. Bold face characters denote tensors. The order of the tensor is implied by the context of the equation.

2.1 Kinematics

A material point in the reference configuration B_0 with position vector $\mathbf{X}$ occupies position $\mathbf{x}$ at time t in the deformed configuration B. Hence we write $\mathbf{x} = \chi(X, t)$. The motion from the original configuration to the deformed configuration shown in Figure 2.1 has a deformation gradient $\mathbf{F}$ given by

$$\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}} \quad , \quad |\mathbf{F}| > 0 \ . \tag{2.1}$$

Applying the polar decomposition theorem to $\mathbf{F}$:

$$\mathbf{F} = \mathbf{V}\mathbf{R} = \mathbf{R}\mathbf{U} \tag{2.2}$$

where $\mathbf{V}$ and $\mathbf{U}$ are the symmetric, positive definite left and right stretch tensors, respectively, and $\mathbf{R}$ is a proper orthogonal rotation tensor. Figure 2.1 illustrates the intermediate orientations defined by the two alternate decompositions of $\mathbf{F}$ defined by Equation 2.2. The determination of $\mathbf{R}$ as defined by Equation 2.2 presents a significant numerical challenge. In Section 3.3, we describe the incremental algebraic algorithm that we use to determine $\mathbf{R}$.

The velocity of the material point $\mathbf{X}$ is written as $\mathbf{v} = \dot{\mathbf{x}}$ where the superposed dot indicates time differentiation holding the material point fixed. The velocity gradient is denoted by $\mathbf{L}$ and may be expressed as

$$\mathbf{L} = \frac{\partial \mathbf{v}}{\partial \mathbf{x}} = \frac{\partial \mathbf{v}}{\partial \mathbf{X}} \frac{\partial \mathbf{X}}{\partial \mathbf{x}} = \dot{\mathbf{F}}\mathbf{F}^{-1} \ . \tag{2.3}$$

The velocity gradient can be written in terms of its symmetric ($\mathbf{D}$) and antisymmetric ($\mathbf{W}$) parts,

$$\mathbf{L} = \mathbf{D} + \mathbf{W} \ . \tag{2.4}$$

Using the right decomposition from Equation 2.2 in Equation 2.3 gives

$$\mathbf{L} = \dot{\mathbf{R}}\mathbf{R}^T + \mathbf{R}\dot{\mathbf{U}}\mathbf{U}^{-1}\mathbf{R}^T \ . \tag{2.5}$$

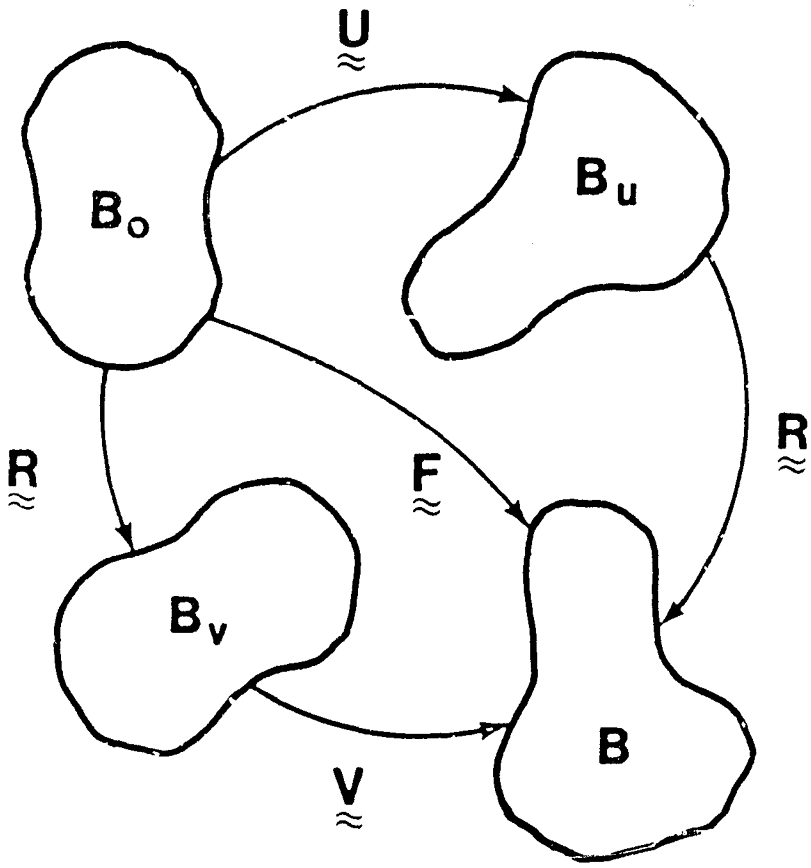

$\underset{\approx}{\mathbf{U}}$
$\mathbf{B}_o$
$\mathbf{B}_u$
$\underset{\approx}{\mathbf{R}}$
$\underset{\approx}{\mathbf{F}}$
$\underset{\approx}{\mathbf{R}}$
$\mathbf{B}_v$
$\mathbf{B}$
$\underset{\approx}{\mathbf{V}}$

Figure 2.1. Original, Deformed and Intermediate Configurations of a Body

Dienes [6] denoted the first term on the right-hand side of Equation 2.5 by $\boldsymbol{\Omega}$:

$$\boldsymbol{\Omega} = \dot{\mathbf{R}}\mathbf{R}^T \ . \tag{2.6}$$

Both $\mathbf{W}$ and $\boldsymbol{\Omega}$ are antisymmetric and represent a rate of rotation (or angular velocity) about some axes. In general, $\boldsymbol{\Omega} \neq \mathbf{W}$. The difference arises when the last term of Equation 2.5 is not symmetric. The symmetric part of $\dot{\mathbf{U}}\mathbf{U}^{-1}$ is the unrotated deformation rate tensor $\mathbf{d}$ as defined below (note that both $\dot{\mathbf{U}}$ and $\mathbf{U}^{-1}$ are symmetric).

$$\mathbf{d} = \frac{1}{2}(\dot{\mathbf{U}}\mathbf{U}^{-1} + \mathbf{U}^{-1}\dot{\mathbf{U}}) = \mathbf{R}^T\mathbf{D}\mathbf{R} \tag{2.7}$$

There are two possible cases which can cause rotation of a material line element: rigid body rotation and shear. Since total shear vanishes along the axes of principal stretch, the rotation of these axes defines the total rigid body rotation of a material point.

It is a simple exercise in vector analysis to show that Equation 2.6 represents the rate of rigid body rotation at a material point (as shown by Dienes [6]). It is equally simple to show that $\mathbf{W}$ represents the rate of rotation of the principal axes of the rate of deformation $\mathbf{D}$. Since $\mathbf{D}$ and $\mathbf{W}$ have no sense of the history of deformation, they are not sufficient to define the rate of rotation in a finite deformation context.

Line elements where the rate of shear vanishes rotate solely due to rigid body rotations. These line elements are along the principal axes of $\dot{\mathbf{U}}$. We will apply a similar observation below as we derive Dienes' [6] expression for calculating $\boldsymbol{\Omega}$.

Using the left decomposition of Equation 2.2 in Equation 2.3 gives

$$\mathbf{L} = \dot{\mathbf{V}}\mathbf{V}^{-1} + \mathbf{V}\boldsymbol{\Omega}\mathbf{V}^{-1} \ . \tag{2.8}$$

Postmultiplying by $\mathbf{V}$ yields an expression which defines the decomposition of $\mathbf{L}$ into $\mathbf{V}$ and $\boldsymbol{\Omega}$:

$$\mathbf{L}\mathbf{V} = \dot{\mathbf{V}} + \mathbf{V}\boldsymbol{\Omega} \ . \tag{2.9}$$

When the dual vector of the above expression is taken, the symmetric $\dot{\mathbf{V}}$ vanishes to yield a set of three linear equations for the three independent components of $\boldsymbol{\Omega}$.

The antisymmetric part of a tensor may be expressed in terms of its dual vector and the permutation tensor e_{ijk}. Define the following dual vectors:

$$\omega_i = e_{ijk}\Omega_{jk} \tag{2.10}$$

$$w_i = e_{ijk}W_{jk} \ . \tag{2.11}$$

Using Equations 2.4, 2.10, and 2.11 in Equation 2.9 results in the expression that Dienes [6] gave for determining $\mathbf{\Omega}$ from $\mathbf{W}$ and $\mathbf{V}$;

$$\boldsymbol{\omega} = \mathbf{w} - 2[\mathbf{V} - \mathbf{I}\,\mathrm{tr}\,\mathbf{V}]^{-1}\mathbf{z} \tag{2.12}$$

where

$$z_i = e_{ijk} V_{jm} D_{mk} \quad . \tag{2.13}$$

We observe from the above expressions that $\mathbf{\Omega} = \mathbf{W}$ if and only if the product $\mathbf{VD}$ is symmetric. This condition requires that the principal axes of the deformation rate $\mathbf{D}$ coincide with the principal axes of the current stretch $\mathbf{V}$. Clearly, a pure rotation is a special case of this condition since $\mathbf{D}$, and consequently Equation 2.13, vanish.

2.1.1 Stress and Strain Rates

Our constitutive model architecture is posed in terms of the conventional Cauchy stress, but we adopt the approach of Johnson and Bammann [7] and define a Cauchy stress in the unrotated configuration. The reader seeking more detail than is presented here should see Flanagan and Taylor [8]. The "true" stress in the deformed configuration is denoted by $\mathbf{T}$. The Cauchy stress in the unrotated configuration is denoted by $\boldsymbol{\sigma}$. These two stress measures are related by

$$\boldsymbol{\sigma} = \mathbf{R}^T\mathbf{T}\mathbf{R} \quad . \tag{2.14}$$

Each material point in the unrotated configuration has its own reference frame which rotates such that the deformation in this frame is a pure stretch. Then $\mathbf{T}$ is simply the tensor $\boldsymbol{\sigma}$ in the fixed global reference frame. The conjugate strain rate measures to $\mathbf{T}$ and $\boldsymbol{\sigma}$ are $\mathbf{D}$ and $\mathbf{d}$, respectively. These strain rates were defined by Equations 2.4 and 2.7, respectively.

The Principal of Material Frame Indifference (or objectivity) stipulates that a constitutive law must be insensitive to a change of reference frame [9]. This requires that only objective quantities may be used in a constitutive law. An objective quantity is one which transforms in the same manner as the energy conjugate stress and strain rate pair under a superposed rigid body motion. The fundamental advantage of the unrotated stress over the true stress is that the material derivative of $\boldsymbol{\sigma}$ is objective, whereas the material derivative of $\mathbf{T}$ is not.

The Jaumann rate defined below is frequently used in constitutive relationships to resolve the need for an objective rate of Cauchy stress.

$$\hat{\mathbf{T}} = \dot{\mathbf{T}} - \mathbf{W}\mathbf{T} + \mathbf{T}\mathbf{W} \quad . \tag{2.15}$$

It is an easy task to show that the Jaumann rate is objective.

A similar stress rate, called the Green-Naghdi rate by Johnson and Bammann [7], can be derived by transforming the rate of the unrotated Cauchy stress to the fixed global frame as follows:

$$\hat{\sigma} = \mathbf{R}\dot{\sigma}\mathbf{R}^T = \dot{\mathbf{T}} - \Omega\mathbf{T} + \mathbf{T}\Omega \ . \tag{2.16}$$

The Jaumann rate and the Green-Naghdi rate are very similar in form. The important difference between the two is that the Green-Naghdi rate is kinematically consistent with the rate of Cauchy stress, while the Jaumann rate is not. By this statement we mean that $\hat{\sigma}$ is identical to $\dot{\mathbf{T}}$ in the absence of rigid body rotations. It is clear that $\hat{\mathbf{T}}$ need not equal $\dot{\mathbf{T}}$ under the same conditions since $\mathbf{W}$ need not vanish with rigid body rotations.

A distinct advantage of the unrotated reference frame is that all constitutive models are cast without regard to finite rotations. This greatly simplifies the numerical implementation of new constitutive models. The rotations of global state variables (e.g., stress and strain) are dealt with on a global level which insures that all constitutive models are consistent. Internal state variables (e.g., backstress) see no rotations whatsoever.

The drawback to working in the unrotated reference frame is that we must accurately determine the rotation tensor, $\mathbf{R}$, which is not a straightforward numerical calculation. We present an incremental, algebraic algorithm to accomplish this task in Section 3.3.

2.1.2 Fundamental Equations

The equations of motion for the body are the momentum equations

$$\nabla \cdot \mathbf{T} - \rho\ddot{\mathbf{u}} + \rho\mathbf{f}_B = 0 \ . \tag{2.17}$$

where ρ is the mass density per unit volume, $\ddot{\mathbf{u}}$ is the acceleration of the material point, and $\mathbf{f}_B$ is a specific (force per mass) body force vector.

We seek the solution to Equation 2.17 subject to the boundary conditions

$$\mathbf{u} = \mathbf{f}(t) \quad \text{on } S_u \tag{2.18}$$

where S_u represents the portion of the boundary on which kinematic quantities are specified (displacement, velocity, and acceleration). In addition to satisfying the kinematic boundary conditions given by Equation 2.18, we must satisfy the traction boundary conditions

$$\mathbf{T} \cdot \mathbf{n} = \mathbf{s}(t) \quad \text{on } S_T \tag{2.19}$$

where S_T represents the portion of the boundary on which tractions are specified. The boundary of the body is given by the union of S_u and S_T, and we note that for a valid mechanics problem, S_u and S_T have a null intersection.

The jump conditions at all contact discontinuities must satisfy the relation

$$(\mathbf{T}^+ + \mathbf{T}^-)\cdot\mathbf{n} = 0 \quad \text{on} \quad S_c \tag{2.20}$$

where S_c represents the contact surface intersection and the superscripts "+" and "−" denote different sides of the contact surface.

The Lagrangian form of the continuity equation is written as

$$\dot{\rho} - \rho \operatorname{tr} \mathbf{D} = 0 \; . \tag{2.21}$$

This is satisfied trivially in our formulation since we do not allow mass transport. Equation 2.21 degenerates to

$$\rho V = \rho_0 V_0 \tag{2.22}$$

where V is the volume and the subscript "0" denotes a reference configuration.

The conservation of energy principle equates the increase in internal energy per unit volume to the rate at which work is being done by the stresses plus the rate at which heat is being added. In the absence of heat conduction

$$\dot{E}_v = \rho \frac{\partial E_m}{\partial t} = \boldsymbol{\sigma} : \mathbf{d} + \rho \dot{Q} \tag{2.23}$$

where E_v is the energy per unit volume, E_m is the energy per unit mass, and $\dot{Q}$ is the heat rate per unit mass. The stress $\boldsymbol{\sigma}$ and the strain rate $\mathbf{d}$ were discussed in the Section 2.2.

3. NUMERICAL FORMULATION

In this chapter, we describe the finite element formulation of the problem and the numerical algorithms required to perform the spatial and temporal integration of the equations of motion.

3.1 Eight-Node Uniform Strain Element

The 8-node three-dimensional isoparametric element is widely used in computational mechanics. The determination of optimal integration schemes for this element, however, presents a difficult dilemma. A one-point integration of the element under-integrates the element resulting in a rank deficiency which manifests itself in spurious zero energy modes, commonly referred to as hourglass modes. A two-by-two-by-two integration of the element over-integrates the element and can lead to serious problems of element locking in fully plastic and incompressible problems. The eight-point integration also carries a tremendous computational penalty compared to the one-point rule. We use the one-point integration of the element and implement an hourglass control scheme to eliminate the spurious modes. The development presented below follows directly from Flanagan and Belytschko [10]. We assume that the reader is familiar with the finite element method and will not go into a complete description of the method. The reader can consult numerous texts on the method, for example Reference [11].

The hexahedral element relates the spatial coordinates x_i to the nodal coordinates x_{iI} through the isoparametric shape functions ϕ_I as follows:

$$x_i = x_{iI}\phi_I(\xi, \eta, \zeta) \tag{3.1}$$

In accordance with indicial notation convention, repeated subscripts imply summation over the range of that subscript. The lowercase subscripts have a range of three corresponding to the spatial coordinate directions. Uppercase subscripts have a range of eight, corresponding to the element nodes.

The same shape functions are used to define the element displacement field in terms of the nodal displacements u_{iI}:

$$u_i = u_{iI}\phi_I \tag{3.2}$$

Since the same shape functions apply to both spatial coordinates and displacements, their material derivative (represented by a superposed dot) must vanish. Hence, the velocity field may be given by

$$\dot{u}_i = \dot{u}_{iI}\phi_I \ , \tag{3.3}$$

Table 3.1. Orthogonal Set of Base Vectors

node	ξ	η	ζ	Σ_I	Λ_{1I}	Λ_{2I}	Λ_{3I}	Γ_{1I}	Γ_{2I}	Γ_{3I}	Γ_{4I}
1	−.5	−.5	−.5	1	−1	−1	−1	1	1	1	−1
2	.5	−.5	−.5	1	1	−1	−1	1	−1	−1	1
3	.5	.5	−.5	1	1	1	−1	−1	−1	1	−1
4	−.5	.5	−.5	1	−1	1	−1	−1	1	−1	1
5	−.5	−.5	.5	1	−1	−1	1	−1	−1	1	1
6	.5	−.5	.5	1	1	−1	1	−1	1	−1	−1
7	.5	.5	.5	1	1	1	1	1	1	1	1
8	−.5	.5	.5	1	−1	1	1	1	−1	−1	−1

and likewise for the acceleration field

$$\ddot{u}_i = \ddot{u}_{iI}\phi_I \quad . \tag{3.4}$$

The velocity gradient tensor, **L**, is defined in terms of nodal velocities as

$$L_{ij} = \dot{u}_{i,j} = \dot{u}_{iI}\phi_{I,j} \quad . \tag{3.5}$$

By convention, a comma preceding a lowercase subscript denotes differentiation with respect to the spatial coordinates (e.g., $\dot{u}_{i,j}$ denotes $\frac{\partial \dot{u}_i}{\partial x_j}$).

The 3-D isoparametric shape functions map the unit cube in ξ_i-space (ξ_i is written explicitly as (ξ, η, ζ)) to a general hexahedron in x_i-space, as shown in Figure 3.1. We choose to center the unit square at the origin in ξ_i space so that the shape functions may be conveniently expanded in terms of an orthogonal set of base vectors, given in Table 3.1, as follows:

$$\phi_I = \frac{1}{8}\Sigma_I + \frac{1}{4}\xi\Lambda_{1I} + \frac{1}{4}\eta\Lambda_{2I} + \frac{1}{4}\zeta\Lambda_{3I} + \frac{1}{2}\eta\zeta\Gamma_{1I} + \frac{1}{2}\zeta\xi\Gamma_{2I} + \frac{1}{2}\xi\eta\Gamma_{3I} + \frac{1}{2}\xi\eta\zeta\Gamma_{4I} \tag{3.6}$$

The above vectors represent the displacement modes of a unit cube. The first vector, Σ_I, accounts for rigid body translation. We call Σ the summation vector since it may be employed in indicial notation to represent the algebraic sum of a vector.

The linear base vectors Λ_{iI} may be readily combined to define three uniform normal strains and three rigid body rotation modes for the unit cube. We refer to Λ_{iI} as the volumetric base vectors since, as we will illustrate below, they are the only base vectors which appear in the element volume expression.

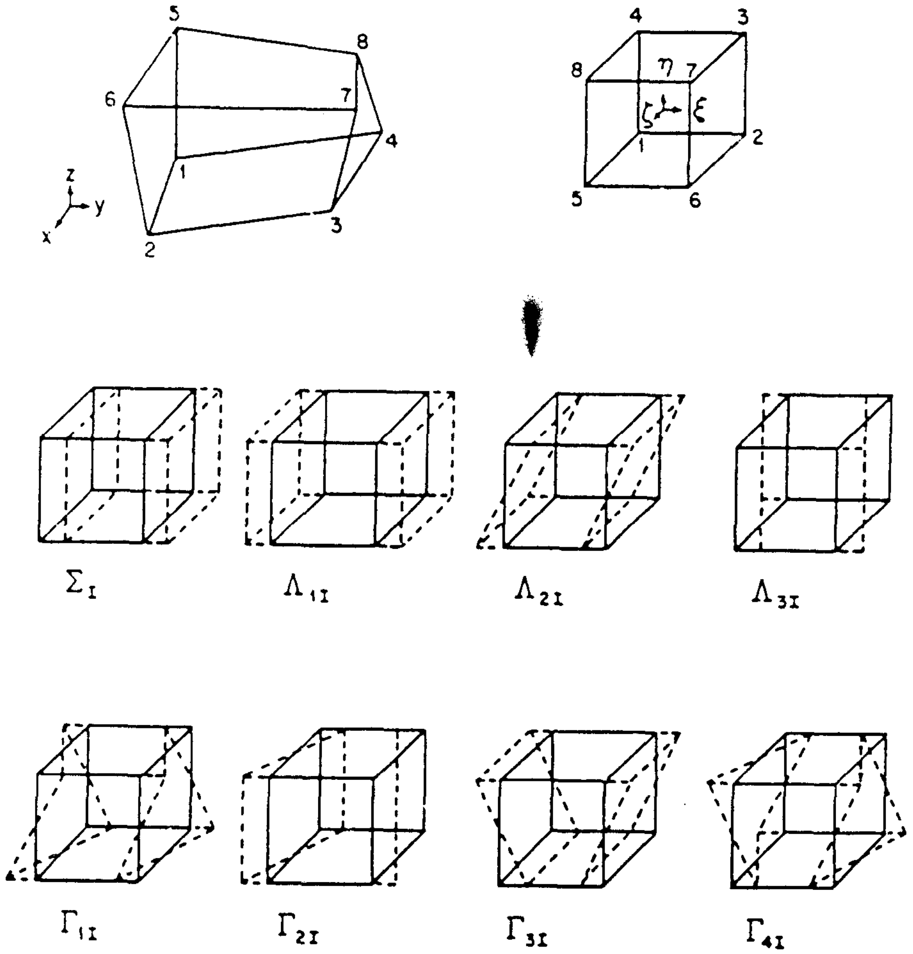


Figure 3.1. Mode Shapes for the Eight-Node Constant Strain Hexahedral Element

The last four vectors, $\Gamma_{\alpha I}$, (Greek subscripts have a range of four) give rise to linear strain modes which are neglected in the uniform strain integration. These vectors define the hourglass patterns for a unit cube. Hence, we refer to $\Gamma_{\alpha I}$ as the hourglass base vectors. The displacement modes represented by the vectors in Table 3.1 are also shown in Figure 3.1.

In the finite element method, we replace the momentum balance (Equation 2.17) with a weak form of the equation. Using the principle of virtual work, we write the weak form of the equation as

$$\sum_e \int_{V_e} (T_{ij,j} + \rho b_i - \rho \ddot{u}_i) \delta u_i dV = 0 \tag{3.7}$$

where δu_i represents an arbitrary virtual displacement field, with the same interpolation as Equation 3.2, which satisfies the kinematic constraints. Integrating by parts and applying Gauss' divergence theorem to Equation 3.7 then gives

$$\sum_e \left[\int_{S_e} T_{ij} n_j \delta u_i dA - \int_{V_e} T_{ij} \delta u_{i,j} dV + \int_{V_e} \rho b_i \delta u_i dV - \int_{V_e} \rho \ddot{u}_i \delta u_i dV \right] = 0 \ . \tag{3.8}$$

The summation symbol represents the assembly of element force vectors into a global nodal force array. We assume that the reader understands the details of this assembly; we will not discuss it further in this document.

The second integral in the preceding equation is used to define the element internal force vector f_{iI} as

$$\delta u_{iI} f_{iI} = \int_{V_e} T_{ij} \delta u_{i,j} dV \ . \tag{3.9}$$

The first and third integrals define the external force vector, and the fourth integral defines the inertial response.

We perform one-point integration by neglecting the nonlinear portion of the element displacement field, thereby considering a state of uniform strain and stress. The preceding expression is approximated by

$$f_{iI} = \bar{T}_{ij} \int_{V_e} \phi_{I,j} dV \tag{3.10}$$

where we have eliminated the arbitrary virtual displacements, and $\bar{T}_{ij}$ represents the assumed uniform stress field which will be referred to as the mean stress tensor. By neglecting the nonlinear displacements, we have assumed that the mean stresses depend only on the mean strains. Mean kinematic quantities are defined by integrating over the element as follows:

$$\dot{\bar{u}}_{i,j} = \frac{1}{V} \int_{V_e} \dot{u}_{i,j} dV \ . \tag{3.11}$$

We now define the discrete gradient operator as

$$B_{iI} = \int_{V_e} \phi_{I,i} dV \quad . \tag{3.12}$$

The mean velocity gradient, applying Equation 3.5, is given by

$$\dot{\bar{u}}_{i,j} = \frac{1}{V} \dot{u}_{iI} B_{jI} \quad . \tag{3.13}$$

Combining Equations 3.10 and 3.12, we may express the nodal forces by

$$f_{iI} = \bar{T}_{ij} B_{jI} \quad . \tag{3.14}$$

Computing nodal forces with this integration scheme requires evaluation of the gradient operator and the element area. These two tasks are linked since

$$x_{i,j} = \delta_{ij} \tag{3.15}$$

where δ_{ij} is the Kronecker delta. Equations 3.1, 3.12, and 3.15 yield

$$x_{iI} B_{jI} = \int_{V_e} (x_{iI} \phi_I)_{,j} dV = V \delta_{ij} \quad . \tag{3.16}$$

Consequently, the gradient operator may be expressed by

$$B_{iI} = \frac{\partial V}{\partial x_{iI}} \quad . \tag{3.17}$$

To integrate the element area in closed form, we use the Jacobian of the isoparametric transformation to transform to an integral over the unit cube:

$$V = \int_V dV = \int_{-\frac{1}{2}}^{+\frac{1}{2}} \int_{-\frac{1}{2}}^{+\frac{1}{2}} \int_{-\frac{1}{2}}^{+\frac{1}{2}} J d\zeta d\eta d\xi \quad . \tag{3.18}$$

The Jacobian is given in terms of the alternator e_{ijk} as

$$J = e_{ijk} \frac{\partial x}{\partial \xi_i} \frac{\partial y}{\partial \eta_k} \frac{\partial z}{\partial \zeta_k} \quad . \tag{3.19}$$

Therefore, Equation 3.18 can be written as

$$V = x_I y_J z_K C_{IJK} \tag{3.20}$$

where

$$C_{IJK} = e_{ijk} \int_{-\frac{1}{2}}^{+\frac{1}{2}} \int_{-\frac{1}{2}}^{+\frac{1}{2}} \int_{-\frac{1}{2}}^{+\frac{1}{2}} \frac{\partial \phi_I}{\partial \xi_i} \frac{\partial \phi_J}{\partial \eta_j} \frac{\partial \phi_K}{\partial \zeta_k} d\zeta d\eta d\xi \quad . \tag{3.21}$$

Observe that the coefficient array C_{IJK} is identical for all hexahedrons. Furthermore, it possesses the alternator properties as given below:

$$C_{IJK} = C_{JKI} = C_{KIJ} = -C_{IKJ} = -C_{JIK} = -C_{KJI} \tag{3.22}$$

Therefore, applying Equations 3.17 and 3.22 to 3.20 yields the following form for evaluating the B-matrix:

$$B_{iI} = \begin{bmatrix} y_J z_K \\ z_J x_K \\ x_J y_K \end{bmatrix} C_{IJK} \ . \tag{3.23}$$

In light of Equation 3.6, it is evident that evaluating each component of C_{IJK} involves integrating a polynomial which is at most bi-quadratic. However, since we are integrating over a symmetric region, any term with a linear dependence will vanish. The only terms which survive the integration will be the constant, square, double square and triple square terms. Furthermore, the alternator properties cause half of these remaining terms to drop out. The resulting expression for C_{IJK} is

$$C_{IJK} = \frac{1}{192} e_{ijk} (3\Lambda_{iI}\Lambda_{jJ}\Lambda_{kK} + \Lambda_{iI}\Gamma_{kJ}\Gamma_{jK} + \Gamma_{kI}\Lambda_{jJ}\Gamma_{iK} + \Gamma_{jI}\Gamma_{iJ}\Lambda_{kK}) \tag{3.24}$$

The above expression is evaluated using Table 3.1, after which practical formula for computing the B-matrix and volume are developed. Since C_{IJK} has the alternator properties given in Equation 3.22, only 56 (the combination of eight-nodes taken three at a time) distinct nonzero terms are possible. However, the volume must be independent of the selection of node 1, which implies that C_{IJK} is invariant if the nodes are permuted according to Table 3.2. Consequently, only 21 (the combination of seven nodes taken two at a time) terms may be independent. Furthermore, once node 1 is selected, three orientations of the node numbering system are possible, as given by the permutation Table 3.3. Therefore, only seven terms of C_{IJK} need be evaluated.

Seven independent terms of C_{IJK} are listed in Table 3.4. These terms may be evaluated via Equation 3.24 and Table 3.1. Only three of these seven terms do not vanish, as indicated in Table 3.4. All other nonzero terms of C_{IJK} are found by permuting the nodes according to Table 3.2 and using the alternator properties of Equation 3.22. Alternatively, the nonzero terms may be generated by applying antisymmetry, $C_{IJK} = -C_{IKJ}$, to Table 3.4, then permuting according to Tables 3.3 and 3.2, successively. The latter scheme straightforwardly results in formula for computing the B-matrix.

The first term of B_{iI} is expressed as

$$B_{11} = \frac{1}{12}[y_2((z_6 - z_3) - (z_4 - z_5)) + y_3(z_2 - z_4)$$

$$+y_4((z_3 - z_8) - (z_5 - z_2)) + y_5((z_8 - z_6) - (z_2 - z_4))$$
$$+y_6(z_5 - z_2) + y_8(z_4 - z_5)] \tag{3.25}$$

Other terms of B_{iI} are evaluated using the same formula after permuting the nodes according to Table 3.2 and, subsequently, permuting the coordinate axes according to Table 3.5. The element volume is most easily computed by contracting the B-matrix and nodal coordinates as per Equation 3.16.

Table 3.2. Nodal Permutations

1	2	3	4	5	6	7	8
2	3	4	1	6	7	8	5
3	4	1	2	7	8	5	6
4	1	2	3	8	5	6	7
5	8	7	6	1	4	3	2
6	5	8	7	2	1	4	3
7	6	5	8	3	2	1	4
8	7	6	5	4	3	2	1

Table 3.3. Three Possible Orientations of Node Numbering

1	2	3	4	5	6	7	8
1	4	8	5	2	3	7	6
1	5	6	2	4	8	7	3

3.1.1 Lumped Mass Matrix

In order to reap the benefits of an explicit architecture, we must diagonalize the mass matrix. We do this by integrating the inertial energy variation as follows:

$$\int_{V_e} \rho u_i o u_i dV = \ddot{u}_{iI} m_{IJ} \delta u_{iJ} \tag{3.26}$$

where

$$m_{IJ} = \rho V \delta_{IJ} \tag{3.27}$$

Table 3.4. Nonzero Terms Generated by Applying Asymmetry

I	J	K	C_{IJK}
1	2	3	$-\frac{1}{12}$
1	2	5	$+\frac{1}{12}$
1	2	6	$+\frac{1}{12}$
1	2	7	0
1	2	8	0
1	3	5	0
1	3	6	0

Table 3.5. Coordinate Axes Permutations

1	2	3
2	3	1
3	1	2

and δ_{IJ} is the Kroneker delta. Clearly the assembly process for the global mass matrix from the individual element matrices results in a global mass matrix which is diagonal and can be expressed as a vector, M_I .

3.2 Explicit Time Integration

PRONTO uses a modified central difference scheme to integrate the equations of motion through time. By this we mean that the velocities are integrated with a forward difference, while the displacements are integrated with a backward difference. The integration scheme for a node is expressed as:

$$\ddot{\mathbf{u}}_t = (\mathbf{f}_t^{EXT} - \mathbf{f}_t^{INT})/\mathbf{M} \tag{3.28}$$

$$\dot{\mathbf{u}}_{t+\Delta t} = \dot{\mathbf{u}}_t + \Delta t \ddot{\mathbf{u}}_t \tag{3.29}$$

$$\mathbf{u}_{t+\Delta t} = \mathbf{u}_t + \Delta t \dot{\mathbf{u}}_{t+\Delta t} \tag{3.30}$$

where $\mathbf{f}_t^{EXT}$ and $\mathbf{f}_t^{INT}$ are the external and internal nodal forces, respectively, $\mathbf{M}$ is the nodal point lumped mass, and Δt is the time increment.

The central difference operator is conditionally stable. It can be shown that the Courant stability limit for the operator with no damping is given in terms of the highest eigenvalue in the system [11]:

$$\Delta t \leq \frac{2}{\omega_{max}} \tag{3.31}$$

In Section 3.5, we discuss how the highest eigenvalue is approximated and how we determine a stable time increment.

3.3 Finite Rotation Algorithm

We stated in Section 2.2 that one of our fundamental numerical challenges in the development of an accurate algorithm for finite rotations was the determination of $\mathbf{R}$, the rotation tensor defined by the polar decomposition of the deformation gradient $\mathbf{F}$. We developed an incremental algorithm for reasons of computational efficiency and numerical accuracy. The validity of the unrotated reference frame is based on the orthogonal transformation given by Equation 2.14. Therefore the crux of integrating Equation 2.6 for $\mathbf{R}$ is to maintain the orthogonality of $\mathbf{R}$. If one integrates $\mathbf{R} = \mathbf{\Omega R}$ via a forward difference scheme, the orthogonality of $\mathbf{R}$ degenerates rapidly no matter how fine the time increments. We instead adapted the algorithm of Hughes and Winget [12] for integrating incremental rotations as follows.

A rigid body rotation over a time increment Δt may be represented by

$$\mathbf{x}_{t+\Delta t} = \mathbf{Q}_{\Delta t}\mathbf{x}_t \tag{3.32}$$

where $\mathbf{Q}_{\Delta t}$ is a proper orthogonal tensor with the same rate of rotation as $\mathbf{R}$ given by Equation 2.6. The total rotation $\mathbf{R}$ is updated via the highly accurate expression below.

$$\mathbf{R}_{t+\Delta t} = \mathbf{Q}_{\Delta t}\mathbf{R}_t \tag{3.33}$$

For a constant rate of rotation, the midpoint velocity and the midpoint coordinates are related by

$$\frac{1}{\Delta t}(\mathbf{x}_{t+\Delta t} - \mathbf{x}_t) = \frac{1}{2}\boldsymbol{\Omega}(\mathbf{x}_{t+\Delta t} + \mathbf{x}_t) \ . \tag{3.34}$$

Combining Equations 3.32 and 3.34 yields

$$(\mathbf{Q}_{\Delta t} - \mathbf{I})\mathbf{x}_t = \frac{\Delta t}{2}\boldsymbol{\Omega}(\mathbf{Q}_{\Delta t} + \mathbf{I})\mathbf{x}_t \ . \tag{3.35}$$

Since $\mathbf{x}_t$ is arbitrary in Equation 3.35, it may be eliminated. We then solve for $\mathbf{Q}_{\Delta t}$. The result is

$$\mathbf{Q}_{\Delta t} = \left(\mathbf{I} - \frac{\Delta t}{2}\boldsymbol{\Omega}\right)^{-1}\left(\mathbf{I} + \frac{\Delta t}{2}\boldsymbol{\Omega}\right) \ . \tag{3.36}$$

The accuracy of this integration scheme is dependent upon the accuracy of the midpoint relationship of Equation 3.34. The rate of rotation must not vary significantly over the time increment. Furthermore, Hughes and Winget [12] showed that the conditioning of Equation 3.36 degenerates as $\Delta t\boldsymbol{\Omega}$ grows.

Our complete numerical algorithm for a single time step is shown in Table 3.6.

This algorithm requires that the tensors $\mathbf{V}$ and $\mathbf{R}$ be stored in memory for each element.

3.4 Determination of Effective Moduli

Algorithms for calculating the stable time increment, hourglass control, bulk viscosity, and nonreflecting boundaries require dilatational and shear moduli. In PRONTO we use an algorithm for adaptively determining the effective dilatational and shear moduli of the material.

Since PRONTO uses an explicit integration algorithm, the constitutive response over a time step can be recast a posteriori as a hypoelastic relationship. We approximate this relationship as isotropic. This defines effective moduli, $\hat{\lambda}$ and $\hat{\mu}$ in terms of the hypoelastic stress increment and strain increment as follows:

$$\Delta\sigma_{ij} = \Delta t(\hat{\lambda}d_{kk}\delta_{ij} + 2\hat{\mu}d_{ij}) \tag{3.37}$$

Table 3.6. Finite Rotation Algorithm

1.	Calculate	$\mathbf{D}$ and $\mathbf{W}$
2.	Compute	$z_i = e_{ijk} V_{jm} D_{mk}$ $\boldsymbol{\omega} = \mathbf{w} - 2[\mathbf{V} - \mathbf{I}tr(\mathbf{V})]^{-1}\mathbf{z}$ $\Omega_{ij} = \frac{1}{2} e_{ijk} \omega_k$
3.	Solve	$\left(\mathbf{I} - \frac{\Delta t}{2}\boldsymbol{\Omega}\right) \mathbf{R}_{t+\Delta t} = \left(\mathbf{I} + \frac{\Delta t}{2}\boldsymbol{\Omega}\right) \mathbf{R}_t$
4.	Calculate	$\dot{\mathbf{V}} = (\mathbf{D} + \mathbf{W})\mathbf{V} - \mathbf{V}\boldsymbol{\Omega}$
5.	Update	$\mathbf{V}_{t+\Delta t} = \mathbf{V}_t + \Delta t \dot{\mathbf{V}}$
6.	Compute	$\mathbf{d} = \mathbf{R}^T \mathbf{D} \mathbf{R}$
7.	Integrate	$\dot{\boldsymbol{\sigma}} = \mathbf{f}(\mathbf{d}, \boldsymbol{\sigma})$
8.	Compute	$\mathbf{T} = \mathbf{R}\boldsymbol{\sigma}\mathbf{R}^T$

Equation 3.37 can be rewritten in terms of volumetric and deviatoric parts as

$$\Delta\sigma_{kk} = \Delta t(3\hat{\lambda} + 2\hat{\mu})d_{kk} \tag{3.38}$$

and

$$s_{ij} = \Delta t 2\hat{\mu} e_{ij} \tag{3.39}$$

where

$$s_{ij} = \Delta\sigma_{ij} - \frac{1}{3}\Delta\sigma_{kk}\delta_{ij} \tag{3.40}$$

and

$$e_{ij} = d_{ij} - \frac{1}{3} d_{kk}\delta_{ij} \quad . \tag{3.41}$$

The effective bulk modulus follows directly from Equation 3.38 as

$$3\hat{K} = 3\hat{\lambda} + 2\hat{\mu} = \frac{\Delta\sigma_{kk}}{\Delta t d_{kk}} \quad . \tag{3.42}$$

Taking the inner product of Equation 3.39 with the deviatoric strain rate and solving for the effective shear modulus $2\hat{\mu}$ gives

$$2\hat{\mu} = \frac{s_{ij} e_{ij}}{\Delta t e_{mn} e_{mn}} \quad . \tag{3.43}$$

Using the result of Equation 3.42 with Equation 3.43, we can calculate the effective dilatational modulus $\hat{\lambda} + 2\hat{\mu}$:

$$\hat{\lambda} + 2\hat{\mu} = \frac{1}{3}(3\hat{K} + 2 \cdot (2\hat{\mu})) \tag{3.44}$$

Table 3.7. Special Cases for Effective Moduli

$\Delta t d_{kk} > 10^{-6}$	$\Delta t^2 e_{ij} e_{ij} > 10^{-12}$	$\hat{\lambda} + 2\hat{\mu}$	$2\hat{\mu}$
Yes	Yes	Eq. 3.44	Eq. 3.43
Yes	No	$\lambda_0 + 2\mu_0$	Eq. 3.45
No	Yes	$\lambda_0 + 2\mu_0$	Eq. 3.43
No	No	$\lambda_0 + 2\mu_0$	$\lambda_0 + 2\mu_0$

If the strain increments are insignificant, Equations 3.42 and 3.43 will not yield numerically meaningful results. In this circumstance, PRONTO sets the dilatational modulus to an initial estimate, $\lambda_0 + 2\mu_0$. An initial estimate of the dilatational modulus is, therefore, the only parameter which every constitutive model is required to provide to the time step control algorithm.

In a case where the volumetric strain increment is significant, but the deviatoric increment is not, the effective shear modulus can be estimated by rearranging Equation 3.44 as follows:

$$2\hat{\mu} = \frac{1}{2}(3(\lambda_0 + 2\mu_0) - 3\hat{K}) \tag{3.45}$$

If neither strain increment is significant, PRONTO sets the effective shear modulus equal to the initial dilatational modulus.

The algorithm that PRONTO uses to estimate the effective dilatational and shear moduli is summarized in Table 3.7. Note that either of effective moduli calculated via this algorithm may be zero or negative. These degenerate cases must be taken into account whenever these moduli are used.

3.5 Determination of the Stable Time Increment

Flanagan and Belytschko [13] provided eigenvalue estimates for the uniform strain hexahedron described in Section 3.1. They showed that the maximum eigenvalue was bounded by

$$8\frac{\lambda + 2\mu}{\rho} \cdot \frac{B_{iI}B_{iI}}{V^2} \geq \omega^2_{max} \geq \frac{8}{3}\frac{\lambda + 2\mu}{\rho}\frac{B_{iI}B_{iI}}{V^2} \quad . \tag{3.46}$$

Using the effective dilatational modulus from Section 3.4 with the eigenvalue estimates of Equation 3.46 allows us to write the stability criteria of Equation 3.31 as

$$\Delta\hat{t}^2 \leq \frac{1}{2}\frac{(\rho_0 V_0)V}{(\lambda + 2\mu)B_{iI}B_{iI}} \quad . \tag{3.47}$$

The stable time increment is determined from Equation 3.47 as the minimum over all elements.

Equation 3.47 is numerically invalid if the effective dilatational modulus is less than or equal to zero. A negative modulus indicates a strain softening situation (the Damage Model, Section 4.4, is the only currently supported constitutive model which allows strain softening), which renders the central difference operator unconditionally unstable. In practice, however, strain softening is generally short lived, so that the calculations can continue in a stable manner once the softening energy has been dissipated. To aid the user in controlling an unstable strain softening situation, we adjust the effective dilatational modulus with the strain softening scale factor (Appendix A, command 11) as follows:

$$\text{If } \hat{\lambda} + 2\hat{\mu} < 0 \text{ then } \hat{\lambda} + 2\hat{\mu} = \frac{\lambda_0 + 2\mu_0}{(\text{ssft})^2} \tag{3.48}$$

To avoid dividing by zero in Equation 3.47, we then enforce the following condition:

$$\hat{\lambda} + 2\hat{\mu} \geq (\lambda_0 + 2\mu_0) \cdot 10^{-6} \tag{3.49}$$

The estimate of the critical time increment given in the Equation 3.47 is for the case where there is no damping in the system. If we define ϵ as the fraction of critical damping in the highest element mode, the stability criteria of Equation 3.47 becomes

$$\Delta t \leq \Delta \hat{t} \left(\sqrt{1 + \epsilon^2} - \epsilon \right) \quad . \tag{3.50}$$

Conventional estimates of the critical time increment size have been based on the transit time of a dilatational wave over the shortest dimension of an element or zone. For the undamped case this gives

$$\Delta t \leq \frac{l}{c} \tag{3.51}$$

where c is the dilatational wave speed.

There are two fundamental and important differences between the time increment limits given by Equations 3.47 and 3.51. First, our time increment limit is dependent on a characteristic element dimension, which is based on the finite element gradient operator and does not require an ad hoc guess of this dimension. This characteristic element dimension, l, is defined by inspection of Equation 3.47 as

$$l = \frac{1}{2} \frac{V}{\sqrt{B_{iI} B_{iI}}} \quad . \tag{3.52}$$

Second, the sound speed used in the estimate is based on the current response of the material and not on the original elastic sound speed. For materials which experience a reduction in stiffness due to plastic flow, this can result in significant increases in the critical time increment.

It should be noted that the stability analysis performed at each time step predicts the critical time increment for the next step. Our assumption is that the conservativeness of this estimate compensates for any reduction in the stable time increment over a single time step.

3.6 Hourglass Control Algorithm

The mean stress-strain formulation of the uniform strain element considers only a fully linear velocity field. The remaining portion of the nodal velocity field is the so-called hourglass field. Excitation of these modes may lead to severe, unresisted mesh distortion. The hourglass control algorithm described here is taken directly from Flanagan and Belytschko [10]. The method isolates the hourglass modes so that they may be treated independently of the rigid body and uniform strain modes.

A fully linear velocity field for the hexahedron can be described by

$$\dot{u}_i^{LIN} = \bar{\dot{u}}_i + \dot{u}_{i,j}(x_j - \bar{x}_j) \quad . \tag{3.53}$$

The mean coordinates $\bar{x}_i$ correspond to the center of the element and are defined as

$$\bar{x}_i = \frac{1}{8} x_{iI} \Sigma_I \quad . \tag{3.54}$$

The mean translational velocity is similarly defined by

$$\bar{\dot{u}}_i = \frac{1}{8} \dot{u}_{iI} \Sigma_I \quad . \tag{3.55}$$

The linear portion of the nodal velocity field may be expressed by specializing Equation 3.53 to the nodes as follows:

$$\dot{u}_{iI}^{LIN} = \bar{\dot{u}}_i \Sigma_I + \dot{\bar{u}}_{i,j}(x_{jI} - \bar{x}_j \Sigma_I) \tag{3.56}$$

where Σ_I is used to maintain consistent index notation and indicates that $\bar{\dot{u}}_i$ and $\bar{x}_j$ are independent of position within the element. From Equations 3.16 and 3.56, and the orthogonality of the base vectors, it follows that

$$\dot{u}_{iI} \Sigma_I = \dot{u}_{iI}^{LIN} \Sigma_I = 8 \bar{\dot{u}}_i \tag{3.57}$$

and

$$\dot{u}_{iI} B_{jI} = \dot{u}_{iI}^{LIN} B_{jI} = V \dot{\bar{u}}_{i,j} \quad . \tag{3.58}$$

The hourglass field $\dot u_{iI}^{HG}$ may now be defined by removing the linear portion of the nodal velocity field:

$$\dot u_{iI}^{HG} = \dot u_{iI} - \dot u_{iI}^{LIN} \tag{3.59}$$

Equations 3.57 through 3.59 prove that Σ_I and B_{jI} are orthogonal to the hourglass field:

$$\dot u_{iI}^{HG} \Sigma_I = 0 \tag{3.60}$$

$$\dot u_{iI}^{HG} B_{jI} = 0 \tag{3.61}$$

Furthermore, it can be shown that the B matrix is a linear combination of the volumetric base vectors, Λ_{iI}, so Equation 3.61 can be written as

$$\dot u_{iI}^{HG} \Lambda_{iI} = 0 \ . \tag{3.62}$$

Equations 3.60 and 3.62 show that the hourglass field is orthogonal to all the base vectors in Table 3.1 except the hourglass base vectors. Therefore, $\dot u_{iI}^{HG}$ may be expanded as a linear combination of the hourglass base vectors as follows:

$$\dot u_{iI}^{HG} = \frac{1}{\sqrt{8}} \dot q_{i\alpha} \Gamma_{\alpha I} \tag{3.63}$$

The hourglass nodal velocities are represented by $\dot q_{i\alpha}$ above (the leading constant is added to normalize $\Gamma_{\alpha I}$). We now define the hourglass shape vector $\gamma_{\alpha I}$ such that

$$\dot q_i = \frac{1}{\sqrt{8}} \dot u_{iI} \gamma_{\alpha I} \ . \tag{3.64}$$

By substituting Equations 3.56, 3.59, and 3.64 into 3.63, then multiplying by $\Gamma_{\alpha I}$ and using the orthogonality of the base vectors, we obtain the following:

$$\dot u_{iI} \Gamma_{\alpha I} - \dot u_{i,j} x_{jI} \Gamma_{\alpha I} = \dot u_{iI} \gamma_{\alpha I} \tag{3.65}$$

With the definition of the mean velocity gradient, Equation 3.13, we can eliminate the nodal velocities above. As a result, we can compute $\gamma_{\alpha I}$ from the following expression:

$$\gamma_{\alpha I} = \Gamma_{\alpha I} - \frac{1}{V} B_{iI} x_{iJ} \Gamma_{\alpha J} \tag{3.66}$$

The difference between the hourglass base vectors $\Gamma_{\alpha I}$ and the hourglass shape vectors $\gamma_{\alpha I}$ is very important. They are identical if and only if the hexahedron is a right-parallelepiped. For a general shape, $\Gamma_{\alpha I}$ is orthogonal to B_{jI} while $\gamma_{\alpha I}$ is orthogonal to the linear velocity field $\dot u_{iI}^{LIN}$. While $\Gamma_{\alpha I}$ defines the hourglass pattern, $\gamma_{\alpha I}$ is necessary to accurately detect hourglassing.

For the purpose of controlling the hourglass modes, we define generalized forces $Q_{i\alpha}$, which are conjugate to $\dot{q}_{i\alpha}$ so that the rate of work is

$$\dot{u}_{iI} f_{iI}^{HG} = \frac{1}{2} Q_{i\alpha} \dot{q}_{i\alpha} \tag{3.67}$$

for arbitrary $\dot{u}_{iI}$. Using Equation 3.64, it follows that the contribution of the hourglass resistance to the nodal forces is given by

$$f_{iI}^{HG} = \frac{1}{2} Q_{i\alpha} \gamma_{i\alpha} \quad . \tag{3.68}$$

Two types of hourglass resistance are possible: artificial stiffness and artificial damping. In PRONTO 3D we use only the stiffness resistance. In terms of the tuneable stiffness κ, the resistance is given by

$$\dot{Q}_i = \frac{\kappa}{2} 2\hat{\mu} \frac{B_{iI} B_{iI}}{A} \dot{q}_{i\alpha} \tag{3.69}$$

Note that the stiffness expression must be integrated, which further requires that this resistance be stored in a global array.

Observe that the nodal antihourglass forces of Equation 3.68 have the shape of $\gamma_{\alpha I}$ rather than $\Gamma_{\alpha I}$. This fact is essential since the antihourglass forces should be orthogonal to the linear velocity field, so that no energy is transferred to or from the rigid body and uniform strain modes by the antihourglassing scheme.

3.7 Artificial Bulk Viscosity

Artificial viscosity is applied to the numerical solution for two reasons. First is to prevent high velocity gradients from collapsing an element before it has a chance to respond. The second reason is to quiet truncation frequency "ringing".

Ideally, one would like to add viscosity only to the highest mode of the element, but isolating this mode is impractical. The standard technique is to simply add viscosity to the volumetric or "bulk" response. This generates a viscous pressure in terms of the volume strain rate as follows:

$$q = b_1 \rho c l \frac{\dot{V}}{V} - \rho \left(b_2 l \frac{\dot{V}}{V} \right)^2 \tag{3.70}$$

where b_1 and b_2 are coefficients for the linear and quadratic terms, respectively. The quadratic term in Equation 3.70 is more important and is designed to "smear" a shock front across several elements. This term yields a jump in energy as a smeared shock passes, which simulates the shock heating. As a result, the smeared shock front can be propagated as a steady wave.

The linear term is intended to dissipate truncation frequency oscillations. The quadratic term is only applied to compressive strain rates since an element cannot collapse in expansion.

The preceding expression is simplified if we use the undamped stable time increment defined by Equation 3.47 and write

$$\Delta \hat{t} = \frac{l}{c} = \sqrt{\frac{V^2}{2B_{iI}B_{iI}} \cdot \frac{\rho}{\lambda + 2\mu}} \tag{3.71}$$

or

$$\Delta \hat{t} = \sqrt{\frac{m}{\lambda + 2\mu} \cdot \frac{V}{2B_{iI}B_{iI}}} \tag{3.72}$$

where m is the element mass. We now define the factor ϵ such that the quadratic viscosity term vanishes in expansion

$$\epsilon = b_1 - b_2^2 \Delta \hat{t} \min(0, D_{kk}) \ . \tag{3.73}$$

This quantity is required for the damped stability criteria of Equation 3.50. Note that the condition imposed by Equation 3.49 prevents Equation 3.73 from yielding so large a value of ϵ that Equation 3.50 would numerically yield a zero value.

We will show below that ϵ can be used to estimate the fraction of critical damping in the highest element mode. Using Equation 3.73 in Equation 3.72 allows us to write the viscous pressure as

$$q = (b_1 - b_2^2 \Delta \hat{t} D_{kk})(\lambda + 2\mu)\Delta \hat{t} D_{kk} \ . \tag{3.74}$$

The bulk viscosity pressure is appended to the stresses during the internal force calculations to yield the following forces.

$$f_{iI} = q B_{iI} \ . \tag{3.75}$$

The above expression can be expanded using Equations 3.72 and 3.73 to yield

$$f_{iI} = \epsilon \rho c l \frac{1}{V} B_{jJ} B_{iI} \dot{u}_{jJ} \ . \tag{3.76}$$

This form indicates that if B_{iI} is an eigenvector, the modal damping is

$$\epsilon \rho \frac{cV}{l} \tag{3.77}$$

The critical damping estimate of the maximum element frequency is

$$2m\omega = 2\frac{\rho V}{8}\frac{2c}{l} = \rho\frac{cV}{2l} \quad . \tag{3.78}$$

The two expressions above show that ϵ is half the fraction of critical damping in the highest mode.

3.8 Adaptive Element Deletion

The adaptive element deletion option was added to PRONTO 3D to provide the capability to model catastrophic material failure. This option should not be confused with the element block deletion option (Appendix A, command 32) which can be used to remove an entire block of material from the analysis at some predetermined time. The keyword here is "adaptive". PRONTO allows the user to specify criteria which define when the material fails within an element. This criteria is defined at the element level and PRONTO checks every time step to determine whether material failure has occurred.

Currently, the user can define failure in terms of energy per unit volume, vonMises stress, pressure, or maximum principal stress. Also, failure criteria can be defined in terms of any internal state variable. Note that the pressure is positive in compression, $p = -\mathrm{tr}(\sigma)$. The adaptive element death capability requires a very mature user who understands how the material behaves. The capability built into the code is quite general, and it is possible for the user to define a nonsensical failure criteria. PRONTO allows the user to specify the failure in terms of a particular variable, a prescribed value of the variable at failure, and what we refer to as the mode of failure. By mode we mean minimum, maximum, or absolute value.

The adaptive element deletion capability is completely vectorized and does not add any appreciable computational penalty. We define a status array (length=NUMEL) which has a value of one or zero. If an element is "alive", the status array contains a value of one for that element. When PRONTO detects that the element has "died", the value of the status array for that element is reduced to zero over five time steps. PRONTO uses the status array to wipe out any contribution that a deleted element makes in step 1 of Section 8.2. Each deleted element undergoes all the calculations which it would if it were not deleted, but its contributions are not included in the time step control algorithm nor the stress divergence. This is accomplished by a few multiplications of critical results by the status array. If the element is not deleted, the results are multiplied by a one and the results are unchanged. If the element is deleted, the results are multiplied by a zero and the results are neutralized. Hence, the overall cost of this algorithm is a few multiplications per element.

When the element is deleted, its contribution to the nodal point lumped mass is still retained. When all the elements connected to a particular node are deleted, the node then becomes a free nodal mass, whose motion PRONTO continues to calculate.

It is more convenient for post-processing to define the status array exactly opposite to our convention. For this reason, each value is flipped as the status array is written to the post-processing data base. Note that if the adaptive element deletion and/or the element block deletion options are used, the element status array is automatically written to the data base.

4. CONSTITUTIVE MODELS

One of the primary reasons for developing PRONTO was to have a numerical testbed for developing constitutive models. As a result, considerable effort was directed to write a flexible material interface subroutine which allows a constitutive model to be added to the code with minimal effort. The MATINT subroutine in PRONTO allows a constitutive modeler to add a new material model to the program by filling in a handful of numbers in data statements which tell the program how to set up the internal data base for the new model. Consequently, the constitutive modeler does not have to understand the inner workings of PRONTO and does not have to write any format statements or juggle the memory allocation in the code. The comments in the FORTRAN explain in great detail how to add the new model. See Appendix C for the steps to be taken to add a new constitutive model.

Currently there are seven material models in the code. Since models can be added with such ease, this number is expected to increase as applications requiring new materials arise.

All material models are written in terms of the unrotated Cauchy stress, $\boldsymbol{\sigma}$, and the deformation rate in the unrotated configuration, $\mathbf{d}$.

For each of the materials described below, we give a list of the internal state variables used in that particular material model. We also give a list of the material constants which are stored in the PROP array (Appendix B, Section 5.0). In the list of material properties, the items marked with an "*" are material properties which are calculated internally. The remaining material properties are the actual values read from the input data.

The relationship between the material models described in this chapter and the equations of state described in Chapter 5 must be understood in order to properly use the equations of state. We have structured PRONTO so that material models can act as a host to an equation of state. Not all of the constitutive models described in this chapter do so. The equations of state cannot be used except in conjunction with a material model. The equation of state can only calculate the volumetric material response. The hydrodynamic material model (Section 4.7) has only volumetric response and just calls the specified equation of state. The elastic plastic hydrodynamic material model (Section 4.9) uses classical J_2 plasticity theory to determine the deviatoric material response and calls the specified equation of state for the volumetric material response. The other constitutive models in this chapter could be restructured to use

Table 4.1. Sign convention used for pressure in each material model

Model	Tension	Compression
Elastic	positive	negative
Elastic-Plastic	positive	negative
Viscoplastic	positive	negative
Damage	positive	negative
Soils and Crushable Foams	negative	positive
Low Density Foams	positive	negative
Hydrodynamic	negative	positive
Elastic-Plastic Hydrodynamic	negative	positive

an equation of state for the volumetric material response if required.

We have followed the historical convention used for each material model for the sign of a positive pressure. We inherited most of these constitutive models from previous codes and did not wish to change what has come to be accepted conventions for a positive pressure. This means that for some models the pressure is positive in tension and for others it is positive in compression. Table 4.1 shows the convention used for each of the material models described in this chapter.

4.1 Elastic Material, Hooke's Law

A linear elastic material is defined using Hooke's Law. In a rate form, this is written as

$$\dot{\sigma} = \lambda(\mathrm{tr}\,\mathrm{d})\delta + 2\mu\mathrm{d} \qquad (4.1)$$

where λ and μ are the elastic Lame material constants.

This model has no internal state variables.

The PROP array for this material contains the following entries:

	PROP(1)	-	Young's Modulus, E
	PROP(2)	-	Poisson's Ratio, ν
*	PROP(3)	-	λ
*	PROP(4)	-	2μ

4.2 Elastic-Plastic Material with Combined Hardening

The elastic-plastic model is based on a standard vonMises type yield condition and uses combined kinematic and isotropic hardening. This model is widely used in many finite element and finite difference computer programs and the many details of its derivation are scattered throughout the literature. Here, we present the model in detail because we feel that many users of the model are not familiar with its underlying assumptions and numerical approximations.

4.2.1 Basic Definitions and Assumptions

Some definitions and assumptions are outlined here. Referring to Figure 4.1, which shows the yield surface in deviatoric stress space, we define the backstress (the center of the yield surface) by the tensor, $\boldsymbol{\alpha}$. If $\boldsymbol{\sigma}$ is the current value of the stress, we define the deviatoric part of the current stress by

$$\mathbf{S} = \boldsymbol{\sigma} - \tfrac{1}{3}\,\mathrm{tr}\,\boldsymbol{\sigma}\boldsymbol{\delta} \tag{4.2}$$

We define the stress difference measured by subtracting the backstress from the deviatoric stress by

$$\boldsymbol{\xi} = \mathbf{S} - \boldsymbol{\alpha} \tag{4.3}$$

The magnitude of the deviatoric stress difference, R, is defined by

$$R = |\boldsymbol{\xi}| = \sqrt{\boldsymbol{\xi} : \boldsymbol{\xi}} \tag{4.4}$$

where we denote the inner product of second order tensors by $\mathbf{S} : \mathbf{S} = S_{ij}S_{ij}$. Note that if the backstress is zero (isotropic hardening case) the stress difference is equal to the deviatoric part of the current stress, $\mathbf{S}$.

The vonMises yield surface is defined as

$$f(\sigma) = \tfrac{1}{2}\boldsymbol{\xi} : \boldsymbol{\xi} = \kappa^2. \tag{4.5}$$

The vonMises effective stress, $\bar{\sigma}$, is defined by

$$\bar{\sigma} = \sqrt{\tfrac{3}{2}\boldsymbol{\xi} : \boldsymbol{\xi}}\ . \tag{4.6}$$

Since R is the magnitude of the deviatoric stress tensor when $\boldsymbol{\alpha} = 0$, it follows that

$$R = \sqrt{2}\kappa = \sqrt{\tfrac{2}{3}}\bar{\sigma} \tag{4.7}$$

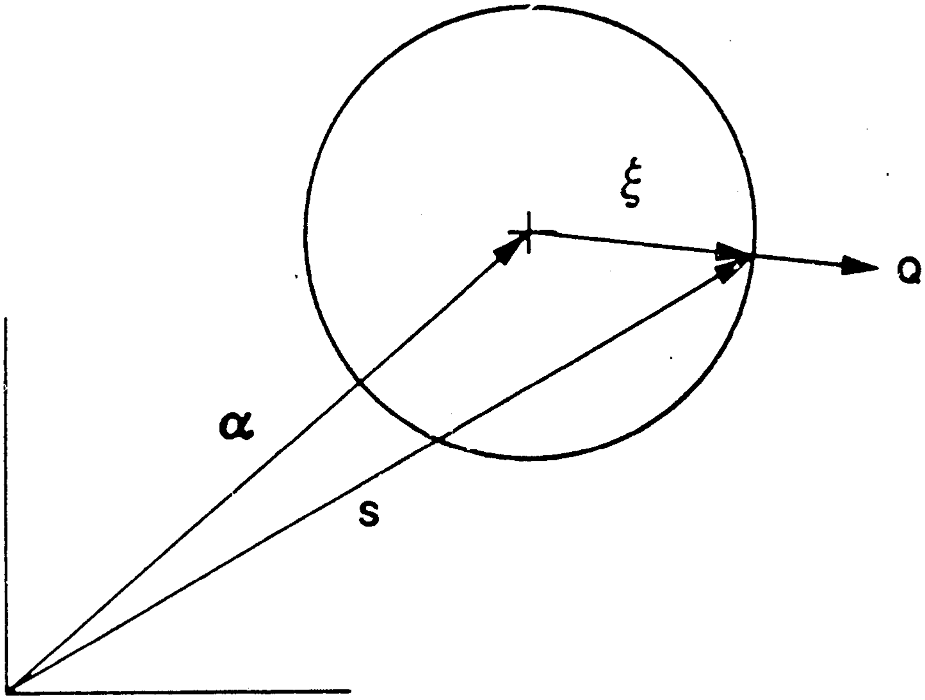


Figure 4.1. Yield Surface in Deviatoric Stress Space

The normal to the yield surface can be determined from Equation 4.5

$$\mathbf{Q} = \frac{\frac{\partial f}{\partial \boldsymbol{\sigma}}}{\left\|\frac{\partial f}{\partial \boldsymbol{\sigma}}\right\|} = \frac{\boldsymbol{\xi}}{R} \tag{4.8}$$

We assume that the strain rate can be decomposed into elastic and plastic parts by an additive decomposition

$$\mathbf{d} = \mathbf{d}^{el} + \mathbf{d}^{pl} \tag{4.9}$$

and assume that the plastic part of the strain rate is given by a normality condition

$$\mathbf{d}^{pl} = \gamma \mathbf{Q}. \tag{4.10}$$

where the scalar multiplier, γ, is to be determined.

A scalar measure of equivalent plastic strain rate is defined by

$$\bar{d}^{pl} = \sqrt{\tfrac{2}{3}\mathbf{d}^{pl} : \mathbf{d}^{pl}} \tag{4.11}$$

which is chosen such that

$$\bar{\sigma}\bar{d}^{pl} = \boldsymbol{\sigma} : \mathbf{d}^{pl}. \tag{4.12}$$

The stress rate is assumed to be purely due to the elastic part of the strain rate and is expressed in terms of Hooke's law by

$$\dot{\boldsymbol{\sigma}} = \lambda(\operatorname{tr}\mathbf{d}^{el})\boldsymbol{\delta} + 2\mu\mathbf{d}^{el} \tag{4.13}$$

where λ and μ are the Lame constants for the material.

Below, we develop the theory for the cases of isotropic hardening, kinematic hardening and combined hardening separately so that the reader can see the details of each case.

4.2.2 Isotropic Hardening

In the isotropic hardening case, the backstress is zero and the stress difference is equal to the deviatoric stress, **S**. We write a consistency condition by taking the rate of Equation 4.5

$$\dot{f}(\boldsymbol{\sigma}) = 2\kappa\dot{\kappa}. \tag{4.14}$$

By "consistency" we mean that the state of stress must remain on the yield surface at all times. We use the chain rule and the definition of the normal to the yield surface given by Equation 4.8 to obtain

$$\dot{f}(\boldsymbol{\sigma}) = \frac{\partial f}{\partial \boldsymbol{\sigma}} : \dot{\boldsymbol{\sigma}} = \left\|\frac{\partial f}{\partial \boldsymbol{\sigma}}\right\| \mathbf{Q} : \dot{\boldsymbol{\sigma}} \tag{4.15}$$

and from Equations 4.4 and 4.5

$$\left\|\frac{\partial f}{\partial \sigma}\right\| = \|\mathbf{S}\| = R \ . \qquad (4.16)$$

Combining Equations 4.14. 4.15. and 4.16

$$\frac{1}{R}\mathbf{S} : \dot{\sigma} = \dot{R} \ . \qquad (4.17)$$

We note that because **S** is deviatoric, $\mathbf{S} : \dot{\sigma} = \mathbf{S} : \dot{\mathbf{S}}$ and

$$\mathbf{S} : \dot{\mathbf{S}} = \frac{d}{dt}\left(\tfrac{1}{2}\mathbf{S} : \mathbf{S}\right) = \frac{d}{dt}\left(\frac{\bar{\sigma}^2}{3}\right) = \tfrac{2}{3}\bar{\sigma}\dot{\bar{\sigma}}. \qquad (4.18)$$

Then Equation 4.17 can be written as

$$\dot{R} = \sqrt{\tfrac{2}{3}}\dot{\bar{\sigma}} = \sqrt{\tfrac{2}{3}}H'\bar{d}^{pl} \qquad (4.19)$$

where H' is the slope of the effective stress versus equivalent plastic strain ($\bar{\sigma}$ versus $\bar{\epsilon}^{pl}$). This is derivable from the data from a uniaxial tension test as shown in Figure 4.2.

The consistency condition. Equation 4.17 and Equation 4.19. result in

$$\sqrt{\tfrac{2}{3}}H'd^{pl} = \mathbf{Q} : \dot{\sigma} \ . \qquad (4.20)$$

We define the trial elastic stress rate $\dot{\sigma}^{tr}$ by

$$\dot{\sigma}^{tr} = \mathbf{C} : \mathbf{d} \qquad (4.21)$$

where C is the fourth-order tensor of elastic coefficients defined by Equation 4.13. Combining the strain rate decomposition defined in Equation 4.9 with Equations 4.20 and 4.21 yields

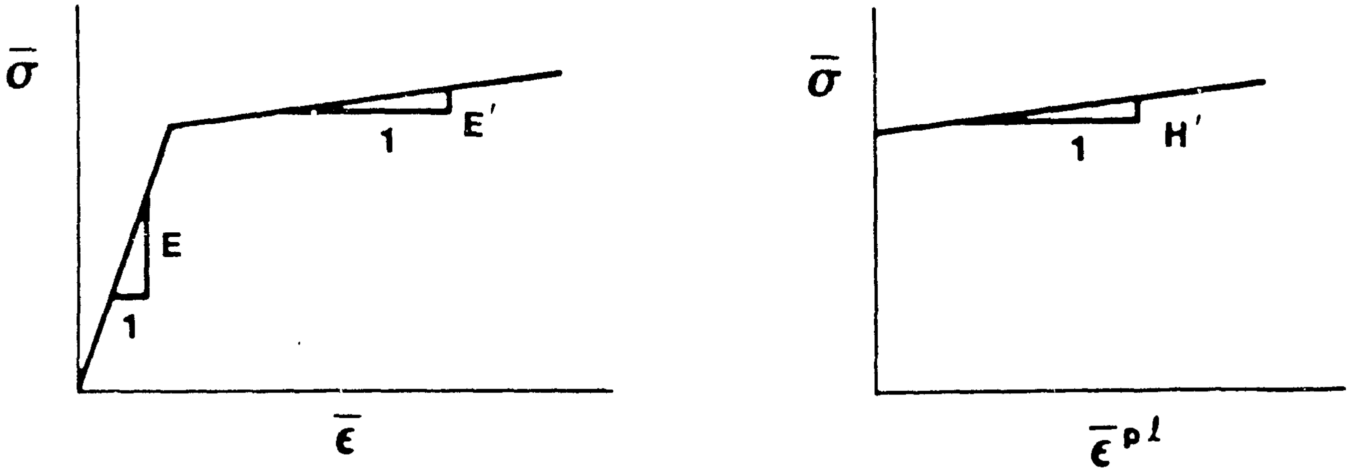
$$\sqrt{\tfrac{2}{3}}H'\bar{d}^{pl} = \mathbf{Q} : \dot{\sigma}^{tr} - \mathbf{Q} : \mathbf{C} : \mathbf{d}^{pl}. \qquad (4.22)$$

We note that because **Q** is deviatoric, $\mathbf{C} : \mathbf{Q} = 2\mu\mathbf{Q}$ and $\mathbf{Q} : \mathbf{C} : \mathbf{Q} = 2\mu$. Then using the normality condition, Equation 4.10, the definition of equivalent plastic strain, Equation 4.11. and Equation 4.22

$$\tfrac{2}{3}H'\gamma = \mathbf{Q} : \dot{\sigma}^{tr} - \gamma 2\mu \qquad (4.23)$$

and since **Q** is deviatoric ($\mathbf{Q} : \dot{\sigma}^{tr} = 2\mu\mathbf{Q} : \mathbf{d}$) we can determine γ from Equation 4.23 as

$$\gamma = \frac{1}{(1 + \frac{H'}{3\mu})}\mathbf{Q} : \mathbf{d}. \qquad (4.24)$$



$$H' = \frac{EE'}{E - E'}$$

Figure 4.2. Conversion of Data From a Uniaxial Tension Test to Equivalent Plastic Strain Versus vonMises Stress

The current normal to the yield surface, $\mathbf{Q}$, and the total strain rate, $\mathbf{d}$, are known quantities. Hence, from Equation 4.24, γ can be determined which can be used in Equation 4.10 to determine the plastic part of the strain rate which, with the additive strain rate decomposition and the elastic stress rate of Equations 4.9 and 4.13, completes the definition of the rate equations.

We still must explain how to integrate the rate equations subject to the constraint that the stress must remain on the yield surface. We will show how that is accomplished in Section 4.2.5.

4.2.3 Kinematic Hardening

Just as before with the isotropic hardening case, we write a vonMises yield condition but now in terms of the stress difference

$$f(\boldsymbol{\xi}) = \tfrac{1}{2}\boldsymbol{\xi} : \boldsymbol{\xi} = \kappa^2. \tag{4.25}$$

It is important to remember that $\boldsymbol{\alpha}$ and $\boldsymbol{\xi}$ are deviatoric tensors. The consistency condition is written for kinematic hardening as

$$\dot{f}(\boldsymbol{\xi}) = 0 \tag{4.26}$$

because the size of the yield surface does not grow with kinematic hardening, therefore, $\dot{\kappa} = 0$. Using the chain rule on Equation 4.26

$$\frac{\partial f}{\partial \boldsymbol{\xi}} : \dot{\boldsymbol{\xi}} = 0 \tag{4.27}$$

and

$$\frac{\partial f}{\partial \boldsymbol{\xi}} = \left\| \frac{\partial f}{\partial \boldsymbol{\xi}} \right\| \mathbf{Q} = R\mathbf{Q} \ . \tag{4.28}$$

Combining Equations 4.27 and 4.28 and assuming that $R \neq 0$

$$\mathbf{Q} : \dot{\boldsymbol{\xi}} = 0 \tag{4.29}$$

or

$$\mathbf{Q} : (\dot{\mathbf{S}} - \dot{\boldsymbol{\alpha}}) = 0 \ . \tag{4.30}$$

A geometric interpretation of Equation 4.30 is shown in Figure 4.3 where it can be seen that the backstress moves in a direction parallel to the normal to the yield surface.

We must now decide how $\dot{\boldsymbol{\alpha}}$ is defined. Recall that for the isotropic hardening case, Equation 4.20

$$\mathbf{Q} : \dot{\boldsymbol{\sigma}} = \sqrt{\tfrac{2}{3}} H' \bar{d}^{pl} = \tfrac{2}{3} H' \dot{\gamma} \ . \tag{4.31}$$

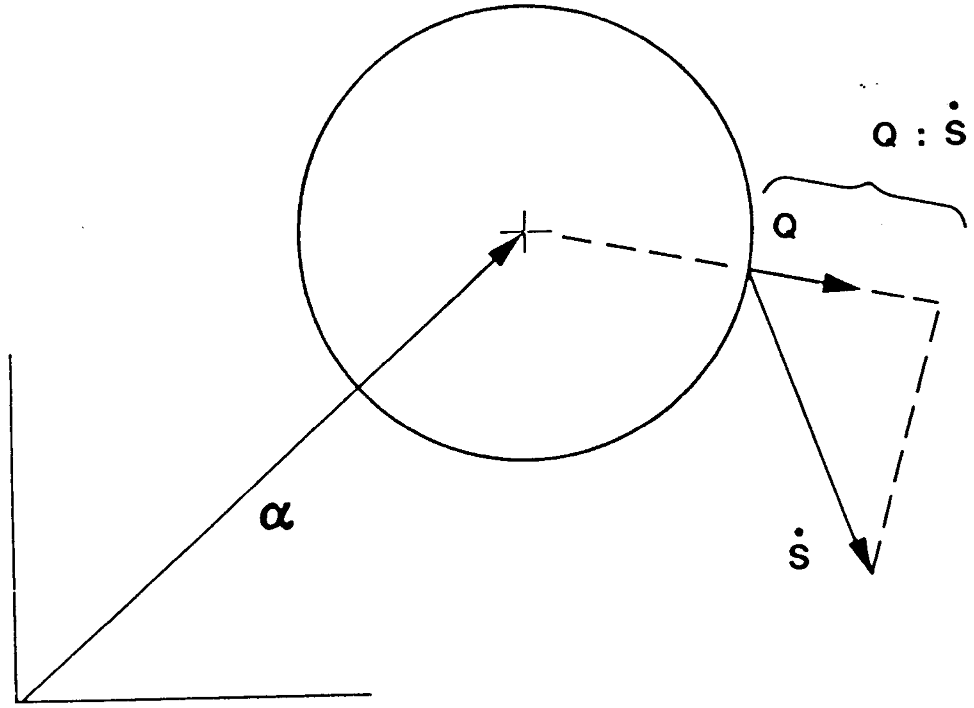


Figure 4.3. Geometric Interpretation of the Consistency Condition for Kinematic Hardening

The kinematic hardening condition assumes that

$$\dot{\boldsymbol{\alpha}} = \phi \mathbf{d}^{pl} = \phi \gamma \mathbf{Q} \tag{4.32}$$

where ϕ is a material parameter. Equation 4.32 combined with Equation 4.30 gives a result identical to the isotropic hardening case, Equation 4.31, if ϕ is chosen to be $\frac{2}{3}H'$. Hence, either Equation 4.31 or 4.32 gives us a scalar condition on $\dot{\boldsymbol{\alpha}}$. Note that both of these are assumptions and must be shown to be reasonable. Of course, experience with material models based on these assumptions has proven them to be reasonable representations of material behavior.

Using Equation 4.31, the strain rate decomposition, Equation 4.9, and the elastic strain rate, Equation 4.13, in the consistency condition for kinematic hardening, Equation 4.30 gives

$$\frac{2}{3}H'\gamma \mathbf{Q} = \dot{\boldsymbol{\sigma}}^{tr} - \mathbf{C} : \mathbf{d}^{pl}. \tag{4.33}$$

Taking the tensor inner product of both sides of Equation 4.33 with $\mathbf{Q}$ gives

$$\mathbf{Q} : \frac{2}{3}H'\gamma \mathbf{Q} = \mathbf{Q} : (\dot{\boldsymbol{\sigma}}^{tr} - 2\mu\gamma \mathbf{Q}) \tag{4.34}$$

Again, because $\mathbf{Q}$ is deviatoric: $\mathbf{C} : \mathbf{Q} = 2\mu \mathbf{Q}$ and $\mathbf{Q} : \mathbf{C} : \mathbf{Q} = 2\mu$.

Solving Equation 4.34 for γ gives

$$\gamma = \frac{1}{(1 + \frac{H'}{3\mu})} \mathbf{Q} : \mathbf{d} \tag{4.35}$$

which is the same result as was obtained for the isotropic hardening case.

4.2.4 Combined Isotropic and Kinematic Hardening

For the combined hardening case we define a scalar parameter, β, which determines the amount of each type of hardening. We require that

$$0 \leq \beta \leq 1 \ . \tag{4.36}$$

Figure 4.4 illustrates the uniaxial response which will be computed for $\bar{\sigma}$ for different choices of β. When $\beta = 0$ we have only kinematic hardening and when $\beta = 1$ we have only isotropic hardening.

We use the results derived before for the independent hardening cases and multiply by the appropriate fraction for each type of hardening. Equations 4.19 and 4.32 are rewritten as

$$\dot{R} = \sqrt{\tfrac{2}{3}} H' \bar{d}^{pl} \beta \tag{4.37}$$

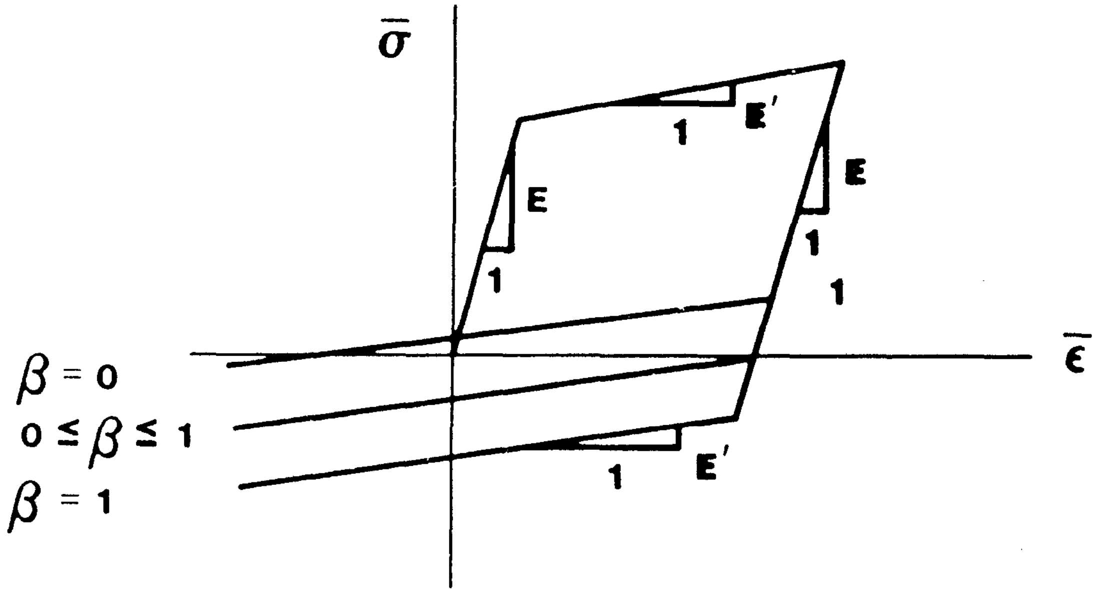


Figure 4.4. Effect of the Choice of the Hardening Parameter, β, on the Computed Uniaxial Response

and

$$\dot{\alpha} = \tfrac{2}{3}H'\mathbf{d}^{pl}(1-\beta) = \tfrac{2}{3}H'\gamma\mathbf{Q}(1-\beta). \tag{4.38}$$

As before, we write a consistency condition

$$\mathbf{Q} : \dot{\boldsymbol{\xi}} = \dot{R} \tag{4.39}$$

or

$$\mathbf{Q} : (\dot{\mathbf{S}} - \dot{\alpha}) = \sqrt{\tfrac{2}{3}}H'\bar{d}^{pl}\beta. \tag{4.40}$$

Using the elastic stress rate and the additive strain rate decomposition with Equation 4.40 and taking the tensor product with the normal, **Q**

$$\mathbf{Q} : \dot{\sigma}^{tr} - \gamma\mathbf{Q} : \mathbf{C} : \mathbf{Q} - \mathbf{Q}\cdot: \left[\tfrac{2}{3}H'\gamma(1-\beta)\right] : \mathbf{Q} = \mathbf{Q} : \left[\sqrt{\tfrac{2}{3}}H'\sqrt{\tfrac{2}{3}}\beta\gamma\right] : \mathbf{Q} \ . \tag{4.41}$$

Solving for γ

$$\gamma = \frac{1}{(1+\frac{H'}{3\mu})}\mathbf{Q} : \mathbf{d} \tag{4.42}$$

which is the same result as we obtained for each of the independent cases.

We summarize the governing equations for the combined theory:

$$\dot{\sigma} = \mathbf{C} : (\mathbf{d} - \mathbf{d}^{pl}) = \dot{\sigma}^{tr} \tag{4.43}$$

$$\dot{R} = \beta\sqrt{\tfrac{2}{3}}H'\bar{d}^{pl} = \beta\tfrac{2}{3}H'\gamma \tag{4.44}$$

$$\dot{\alpha} = (1-\beta)\tfrac{2}{3}H'\mathbf{d}^{pl} \tag{4.45}$$

$$\mathbf{d}^{pl} = \begin{cases} 0, & \text{elastic; if } f(\boldsymbol{\xi}) < \kappa^2 \\ \gamma\mathbf{Q}, & \text{plastic; if } f(\boldsymbol{\xi}) \geq \kappa^2 \end{cases} \tag{4.46}$$

$$\gamma = \frac{1}{(1+\frac{H'}{3\mu})}\mathbf{Q} : \mathbf{d} \tag{4.47}$$

$$\mathbf{Q} = \frac{\frac{\partial f}{\partial \sigma}}{\|\frac{\partial f}{\partial \sigma}\|} = \frac{\boldsymbol{\xi}}{R} \tag{4.48}$$

4.2.5 Numerical Implementation

Our finite element algorithm requires an incremental form of Equations 4.43–4.48. Additionally, we must have an algorithm which integrates the incremental equations subject to the constraint that the stress remains on the yield surface.

The incremental analogs of Equations 4.43 through 4.45 are

$$\sigma_{n+1} = \sigma^{tr}_{n+1} - \Delta\gamma 2\mu \mathbf{Q} \tag{4.49}$$

$$R_{n+1} = R_n + \tfrac{2}{3}\beta H' \Delta\gamma \tag{4.50}$$

and

$$\boldsymbol{\alpha}_{n+1} = \boldsymbol{\alpha}_n + (1-\beta)\tfrac{2}{3}H'\Delta\gamma\mathbf{Q} \tag{4.51}$$

where $\Delta\gamma$ represents the product of the time increment and the equivalent plastic strain rate ($\Delta\gamma = \Delta t\gamma$). The subscripts n and $n+1$ refer to the beginning and end of a time step, respectively.

We also need an incremental analog to the rate forms of the consistency condition given by Equations 4.14, 4.26, and 4.40. At the end of the time step, we insist that the stress state must be on the yield surface. Hence, the incremental consistency condition is

$$\boldsymbol{\alpha}_{n+1} + R_{n+1}\mathbf{Q} = \mathbf{S}_{n+1}. \tag{4.52}$$

Equation 4.52 is shown graphically in Figure 4.5.

Substituting the definitions given by Equations 4.49 through 4.51 into the consistency condition of Equation 4.52

$$\left[\boldsymbol{\alpha}_n + (1-\beta)\tfrac{2}{3}H'\Delta\gamma\mathbf{Q}\right] + \left[R_n + \tfrac{2}{3}\beta H'\Delta\gamma\right]\mathbf{Q} = \left[\mathbf{S}^{tr}_{n+1} - \Delta\gamma 2\mu\mathbf{Q}\right]. \tag{4.53}$$

Taking the tensor product of both sides of Equation 4.53 with $\mathbf{Q}$ and solving for $\Delta\gamma$

$$\Delta\gamma = \frac{1}{2\mu}\frac{1}{(1+\frac{H'}{3\mu})}\left(\|\boldsymbol{\xi}^{tr}_{n+1}\| - R_n\right). \tag{4.54}$$

It follows from Equation 4.54 that the plastic strain increment is proportional to the magnitude of the excursion of the elastic trial stress past the yield surface (see Figure 4.6).

Using the result of Equation 4.54 in Equations 4.49 through 4.51 completes the algorithm. In addition, we can compute

$$\Delta\mathbf{d}^{pl} = \Delta\gamma\mathbf{Q} \tag{4.55}$$

and

$$\Delta\bar{d}^{pl} = \sqrt{\tfrac{2}{3}}\Delta\gamma. \tag{4.56}$$

The results of Equation 4.54 applied to Equation 4.49 show that the final stress is calculated by returning the elastic trial stress radially to the final yield surface at the end of the time step. (Hence the derivation of the name Radial Return Method.) Estimates of the accuracy of this method and other methods for similarly integrating the rate equations are available in Krieg and Krieg [14] and Schreyer, et al. [15]. Note that the last term in Equation 4.49 (the radial return correction) is purely deviatoric.

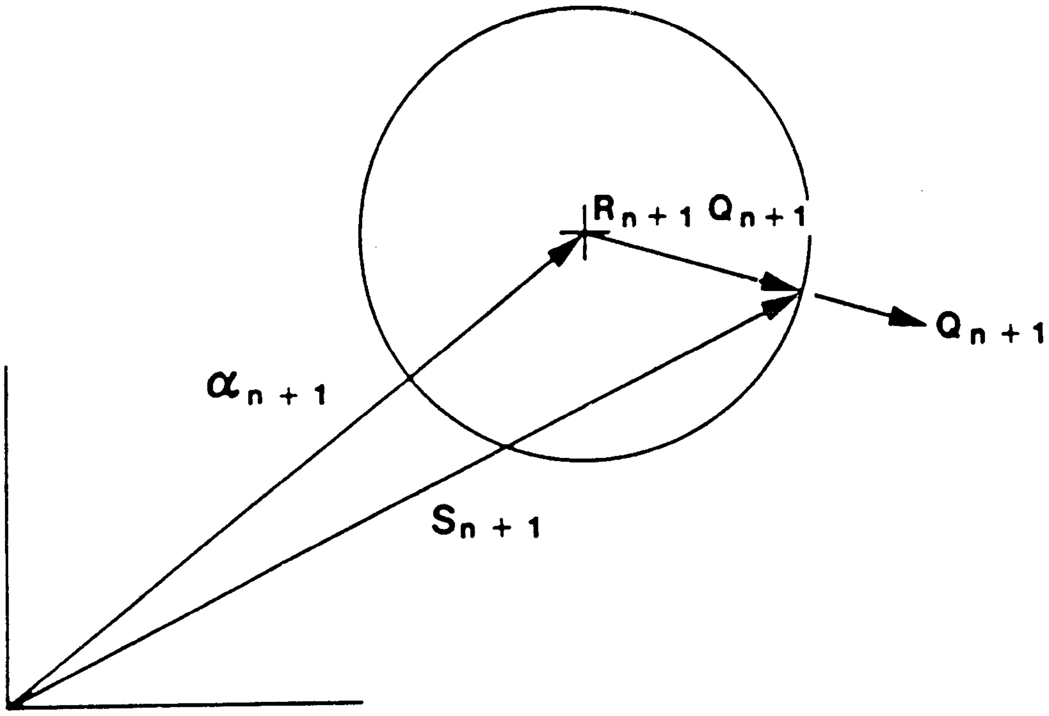


Figure 4.5. Geometric Interpretation of the Incremental Form of the Consistency Condition for Combined Hardening

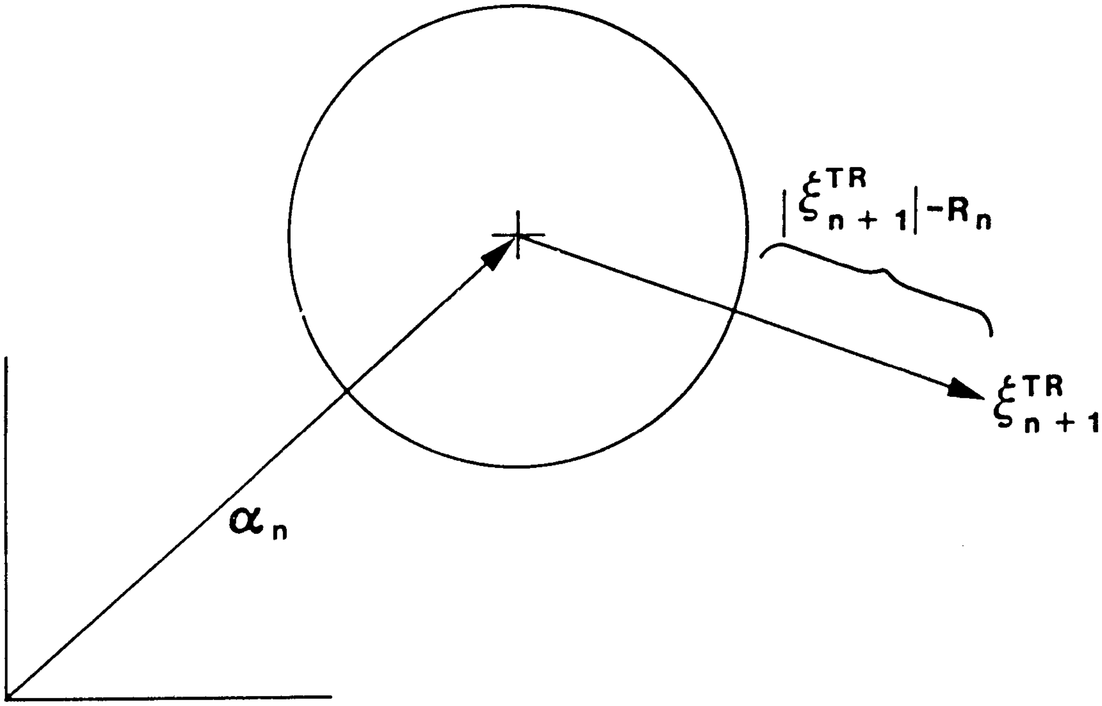


Figure 4.6. Geometric Interpretation of the Radial Return Correction

The elastic-plastic material model uses eight internal state variables:

EQPS -	equivalent plastic strain
RADIUS -	current radius of yield surface
ALPHA11 -	xx component of backstress in unrotated configuration
ALPHA22 -	yy component of backstress in unrotated configuration
ALPHA33 -	zz component of backstress in unrotated configuration
ALPHA12 -	xy component of backstress in unrotated configuration
ALPHA23 -	yz component of backstress in unrotated configuration
ALPHA31 -	zx component of backstress in unrotated configuration

The PROP array for this material contains the following entries:

	PROP(1) -	Young's Modulus, E
	PROP(2) -	Poisson's Ratio, ν
	PROP(3) -	Yield Stress, σ_{yd}
	PROP(4) -	Hardening Modulus, E'
	PROP(5) -	β
*	PROP(6) -	2μ
*	PROP(7) -	3μ
*	PROP(8) -	$1/(2\mu(1 + H'/3\mu)$ (Note:$H' = H/(1 - H/E)$
*	PROP(9) -	λ
*	PROP(10) -	$2\beta H'/3$
*	PROP(11) -	$2(1 - \beta)H'/3$

4.3 Viscoplastic Material Model

The viscoplastic material model presented here represents a simple rate dependent plasticity model. The model is intended for relatively low strain rate ($\|\mathbf{d}\| < 200$) and is not recommended for high rates of impact. More details of the model can be found in Taylor and Becker [16] and Perzyna [17]. The model assumes an additive strain rate decomposition identical to the elastic-plastic model

$$\mathbf{d} = \mathbf{d}^{el} + \mathbf{d}^{pl}. \tag{4.57}$$

The stress rate is assumed to be given by the elastic part of the strain rate using Hooke's law

$$\dot{\sigma} = \mathbf{C} : \mathbf{d}^{el} = \mathbf{C} : (\mathbf{d} - \mathbf{d}^{pl}) \tag{4.58}$$

which can be written more clearly in index notation as

$$\dot{\sigma}_{ij} = \lambda d^{el}_{kk}\delta_{ij} + 2\mu d^{el}_{ij}. \tag{4.59}$$

We define the vonMises equivalent stress by

$$\bar{\sigma} = \sqrt{\tfrac{3}{2}\mathbf{S} : \mathbf{S}} = \sqrt{\tfrac{3}{2}S_{ij}S_{ij}} \tag{4.60}$$

where S is the deviatoric part of $\boldsymbol{\sigma}$.

For this isothermal model, we use isotropic hardening only. Hence, we can write the yield stress as

$$\sigma_0 = \sigma_0(\bar{\epsilon}^{pl}) \tag{4.61}$$

where $\bar{\epsilon}^{pl}$ is the equivalent plastic strain. In this model, we assume isotropic hardening with a Hardening Modulus, E'. This is defined by identifying an equivalent plastic strain rate by

$$\bar{\sigma}\dot{\bar{\epsilon}}^{pl} = \boldsymbol{\sigma} : \mathbf{d}^{pl} \tag{4.62}$$

and

$$\bar{\epsilon}^{pl} = \int_0^t \sqrt{\tfrac{2}{3}\mathbf{d}^{pl} : \mathbf{d}^{pl}}\, dt \ . \tag{4.63}$$

We define the yield function as

$$f(\boldsymbol{\sigma}, \sigma_0) = \bar{\sigma} - \sigma_0(\bar{d}^{pl}) \ . \tag{4.64}$$

The plastic strain rate is assumed to be given by a stress potential as

$$\mathbf{d}^{pl} = \frac{\partial g(\boldsymbol{\sigma})}{\partial \boldsymbol{\sigma}} \tag{4.65}$$

and we assume an associated flow rule which implies that

$$g(\boldsymbol{\sigma}) = \bar{g}(f) = g(\bar{\sigma}, \sigma_0) \ . \tag{4.66}$$

Then Equation 4.65 can be written as

$$\mathbf{d}^{pl} = \frac{\partial g}{\partial \bar{\sigma}}\frac{\partial \bar{\sigma}}{\partial \boldsymbol{\sigma}} \ . \tag{4.67}$$

We use a power law for $\frac{\partial g}{\partial \bar{\sigma}}$

$$\frac{\partial g}{\partial \bar{\sigma}} = \begin{cases} \gamma\left(\frac{\bar{\sigma}}{\sigma_0} - 1\right)^p & \bar{\sigma} \geq \sigma_0 \\ 0 & \bar{\sigma} < \sigma_0 \end{cases} \tag{4.68}$$

where γ and p are material parameters.

Equation 4.68 indicates that the plastic strain rate is proportional to the overstress above the current value of the yield stress. Hence, the higher the overstress, the greater the plastic strain, which leads to a reduction in the stress rate given by Equation 4.58 and an increase in strain hardening given by Equation 4.61.

Consider a uniaxial tension test. The solid curve shown in Figure 4.7 shows the locus of apparent yield strengths $\hat{\sigma}_y$ for mild steel at different strain rates. The apparent yield strength is the measured yield strength for a specimen from a tension test at a given strain rate. At different strain rates, different yield strengths are found. If the elastic strain rate is assumed negligible, using Equation 4.67 and 4.68, the uniaxial strain rate is

$$d = \gamma \left(\frac{\bar{\sigma}}{\sigma_0} - 1 \right)^p . \tag{4.69}$$

Solving Equation 4.69 for the vonMises equivalent stress gives a relation for the apparent yield stress

$$\hat{\sigma}_y = \sigma_0 \left[1 + \left(\frac{d}{\gamma} \right)^{\frac{1}{p}} \right] . \tag{4.70}$$

If the rate of deformation is very slow (e.g., $d \rightarrow 0$), or the fluidity constant is very large (e.g., $\gamma \rightarrow \infty$), then the yield stress given by Equation 4.70 is equal to the static yield stress and the static yield condition of a rate-independent constitutive theory is satisfied. If the motion is very rapid (e.g., $d \rightarrow \infty$), or the fluidity constant is zero, the response is elastic, since the value of yield stress is not restricted by Equation 4.70. Values of effective yield stress as a function of strain rate for different choices of the flow parameters, γ and p, are shown in Figure 4.7.

It follows that

$$\bar{d}^{pl} = \frac{\partial g}{\partial \bar{\sigma}} \tag{4.71}$$

and

$$\frac{\partial \bar{\sigma}}{\partial \sigma} = \frac{1}{\bar{\sigma}} \mathbf{S} . \tag{4.72}$$

Using these definitions of the flow potential in Equation 4.65 yields

$$\mathbf{d}^{pl} = \bar{d}^{pl} \frac{1}{\bar{\sigma}} \mathbf{S} . \tag{4.73}$$

The numerical algorithm used in this model consists of a backward difference integration of the rate equations. The algorithm proceeds as shown in Table 4.2.

The viscoplastic material model uses two internal state variables:

EQPS	-	equivalent plastic strain
SIGYLD	-	current value of yield stress

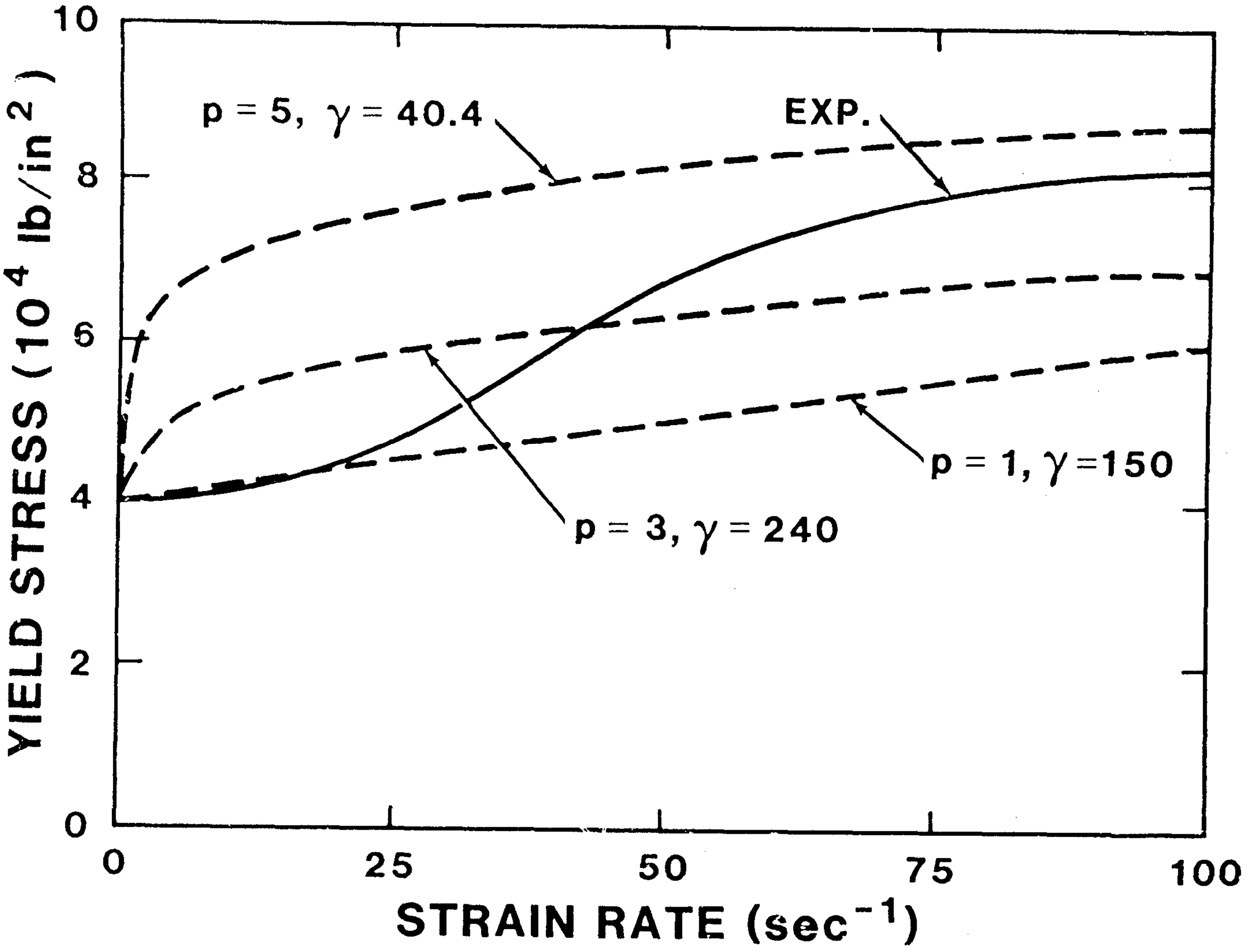


Figure 4.7. Yield Stress as a Function of Strain Rate as Defined by the Viscoplastic Material Model

Table 4.2. Algorithm for Viscoplastic Material Model

```
1. Calculate the elastic trial stress
        $\boldsymbol{\sigma}^{tr}_{n+1} = \boldsymbol{\sigma}_n + \Delta t \mathbf{C}\mathbf{d}$
2. Calculate the equivalent trial stress
        $\bar{\sigma}^{tr} = \sqrt{\frac{3}{2}\mathbf{S}^{tr} : \mathbf{S}^{tr}}$
3. Check for yield
        $\bar{\sigma} - \sigma_0(\bar{d}^{pl}) \leq 0$ ; skip step 4
        $\bar{\sigma} - \sigma_0(\bar{d}^{pl}) > 0$ ; continue below with step 4
4. Yield exceeded, calculate
        $\Delta \mathbf{D}^{pl} = \gamma \left(\frac{\bar{\sigma}^{tr}}{\sigma_0} - 1\right)^p \frac{\Delta t}{\bar{\sigma}} \mathbf{S}^{tr}$
        $\Delta \boldsymbol{\sigma} = \mathbf{C}(\Delta t \mathbf{d} - \Delta \mathbf{d}^{pl}) = \boldsymbol{\sigma}^{tr} - 2\mu \Delta \mathbf{d}^{pl}$
        $\Delta \bar{d}^{pl} = \gamma \left(\frac{\bar{\sigma}^{tr}}{\sigma_0} - 1\right)^p$
        $\sigma_{n+1} = \boldsymbol{\sigma}_n + \Delta \boldsymbol{\sigma}$
        $\sigma_0^{n+1} = \sigma_0(\bar{d}^{pl}_n + \Delta \bar{d}^{pl}_{n+1})$
```

The PROP array for this material contains the following entries:

	PROP(1)	-	Young's Modulus, E
	PROP(2)	-	Poisson's Ratio, ν
	PROP(3)	-	Yield Stress, σ_{yd}
	PROP(4)	-	Hardening Modulus, E'
	PROP(5)	-	γ
	PROP(6)	-	p
*	PROP(7)	-	2μ
*	PROP(8)	-	λ
*	PROP(9)	-	$H' = H/(1 - H/E)$

4.4 Damage Model

The damage model in PRONTO simulates the dynamic fracture behavior of brittle rock. It is based on work started by Kipp and Grady, [18] continued by Taylor, Chen and Kuszmaul [19] [20] and recently modified by B. J. Thorne (first documented here). The essential feature of this model is the treatment of the dynamic fracture process in rock as a continuous accrual of damage in tension, where the damage mechanism is attributed to microcracking in the rock medium. The fundamental assumption of the damage model is that the material is permeated by an array of randomly distributed microcracks which grow and interact with one another under tensile loading.

The compressive response of the material is assumed to be elastic-perfectly-plastic and follows the theory of Section 4.2 with the hardening modulus, H', set to zero. In this section, we present the equations governing the tensile response of the material.

Following the work of Budiansky and O'Connell [21], we write the effective bulk modulus of a cracked medium as

$$\frac{\bar{K}}{K} = 1 - \frac{16}{9}\left(\frac{1-\bar{\nu}^2}{1-2\bar{\nu}}\right) C_d \quad . \tag{4.74}$$

In Equation 4.74, the barred quantities represent degraded or effective quantities in the fractured medium. We denote the undegraded bulk modulus and Poisson's ratio by K and ν, respectively. The crack density, C_d, represents the volume fraction of the material occupied by flaws and is given by

$$C_d = \frac{45}{16}\frac{(\nu-\bar{\nu})(2-\bar{\nu})}{(1-\bar{\nu}^2)[10\nu-\bar{\nu}(1+3\nu)]} \quad . \tag{4.75}$$

A Wiebull distribution is used to determine the number of flaws per unit volume, N, active at a given mean volumetric strain,

$$N = k{\epsilon_v}^m \tag{4.76}$$

where $\epsilon_v = \frac{1}{3}\int tr\mathbf{d}dt$ is the volumetric strain and k and m are material constants.

The nominal fragment size, a, is given by an expression derived by Grady [22]

$$a = \tfrac{1}{2}\left(\frac{\sqrt{20}K_{IC}}{\rho c\dot{\epsilon}_{max}}\right)^{2/3} \tag{4.77}$$

where K_{IC} is the fracture toughness of the material, ρ is the material density, $c = \sqrt{E/\rho}$ is the wave speed and $\dot{\epsilon}_{max}$ is the maximum positive (tensile) mean volumetric strain rate the material has ever experienced. The crack density is proportional to the number of flaws per unit volume and the nominal fragment size,

$$C_d = \beta N a^3, \tag{4.78}$$

where β is a proportionality constant. Combining Equations 4.76 through 4.78 gives an expression for the crack density in terms of the current mean volumetric strain and the maximum previous mean volumetric strain rate, $\dot{\epsilon}_{max}$,

$$C_d = \frac{5k}{2}\left(\frac{K_{IC}}{\rho c\dot{\epsilon}_{max}}\right)^2 \epsilon_v^m \tag{4.79}$$

where we have absorbed the proportionality constant, β, into the material constant, k.

Combining Equations 4.79 and 4.75 gives

$$\frac{5k}{2}\left(\frac{K_{IC}}{\rho c\dot{\epsilon}_{max}}\right)^2 \epsilon_v^m = \frac{45}{16}\frac{(\nu-\bar{\nu})(2-\bar{\nu})}{(1-\bar{\nu}^2)[10\nu-\bar{\nu}(1+3\nu)]} \quad . \tag{4.80}$$

Equation 4.80 gives a relation which can be solved for the effective Poisson's ration of the degraded material. Unfortunately, for given values of C_d and ν, the equation is a cubic in $\bar{\nu}$ and the determination of $\bar{\nu}$ is a nontrivial calculation. As a simplification, Equation 4.80 has been approximated with a linear, analytic function for $\bar{\nu}$ in terms of ν and C_d,

$$\bar{\nu} = \nu\left(1 - \frac{16}{9}C_d\right) \quad . \tag{4.81}$$

The error associated with using Equation 4.81 instead of Equation 4.80 to determine $\bar{\nu}$ is generally less than 5 percent. Once $\bar{\nu}$ is known, it is used in Equation 4.74 to determine the effective bulk modulus of the material. The error of the bulk modulus due to using Equation 4.81 instead of Equation 4.80 is less than 1 percent.

It is convenient to define a damage parameter, D, where $0 \leq D \leq 1$ as

$$D = \frac{16}{9}f_1(\bar{\nu})C_d \tag{4.82}$$

where

$$f_1(\bar{\nu}) = \frac{(1-\bar{\nu}^2)}{(1-2\bar{\nu})} . \tag{4.83}$$

This definition of damage follows directly from inspection of Equation 4.74 and results in an expression for the total mean stress or pressure as

$$p = 3K(1-D)\epsilon_v \quad . \tag{4.84}$$

We assume that the deviatoric response of the material is degraded in a manner consistent with the bulk response by defining a degraded shear modulus, $\bar{\mu}$ such that

$$\mathbf{S} = 2\bar{\mu}\mathbf{e} = \frac{3\bar{K}(1-2\bar{\nu})}{2(1+\bar{\nu})}\mathbf{e} \quad . \tag{4.85}$$

where $\mathbf{S}$ is the deviatoric part of the stress and $\mathbf{e}$ is the deviatoric part of the strain. In addition the yield stress in tension is degraded by taking $\bar{Y} = (1-D)Y$, where Y is the yield stress in compression.

Taking the rate of Equations 4.82 through 4.85;

$$\dot{D} = \frac{16}{9} f_1(\bar{\nu})\dot{C}_d + \frac{16}{9} C_d \dot{f}_1(\bar{\nu}), \tag{4.86}$$

$$\dot{p} = 3K(1 - D)\dot{\epsilon}_v - 3K\epsilon_v\dot{D}, \tag{4.87}$$

$$\dot{\mathbf{S}} = 2\bar{\mu}\dot{\mathbf{e}} - 2\dot{\bar{\mu}}\mathbf{e}, \tag{4.88}$$

where

$$\dot{f}_1(\bar{\nu}) = \frac{\partial f_1}{\partial \bar{\nu}} \frac{\partial \bar{\nu}}{\partial C_d} \dot{C}_d \tag{4.89}$$

and

$$\dot{\bar{\mu}} = \frac{8\bar{\nu}(1 - D)K}{(1 + \bar{\nu})^2}\dot{C}_d - \frac{3K(1 - 2\bar{\nu})}{2(1 + \bar{\nu})}\dot{D}. \tag{4.90}$$

It follows that

$$\dot{D} = \frac{16}{9}\left[f_1(\bar{\nu}) - \frac{16}{9}\nu f_2(\bar{\nu})C_d\right]\dot{C}_d \tag{4.91}$$

where

$$f_2(\bar{\nu}) = \frac{2(1 - \bar{\nu} + \bar{\nu}^2)}{(1 - 2\bar{\nu})^2} \tag{4.92}$$

and

$$\dot{C}_d = \tfrac{5}{2} km \left(\frac{K_{IC}}{\rho c\dot{\epsilon}_{max}}\right)^2 \epsilon^{m-1}\dot{\epsilon}_v. \tag{4.93}$$

Equations 4.87, 4.88, 4.91, and 4.93 represent seven coupled ordinary differential equations to be integrated over each time step. In PRONTO, we use a simple forward difference integration operator.

The damage material model requires six material constants to characterize the material. Young's modulus, Poisson's ratio, the yield stress in compression, and the fracture toughness of the material are all conventional material properties which can be obtained from standard material tests. The remaining two material constants are k and m for the Wiebull distribution of Equation 4.76. The parameter m relates tensile fracture to strain rate and has been determined by Kuszmaul [23] to have a value of 6. A definition of the parameter, k, has not been found. Fortunately, the model is not particularly sensitive to the value of k, so it is reasonable to define k for the special case of a Poisson's ratio of zero. For the special case of $\nu = 0$ and a constant strain rate, the definition of k is exactly the same as derived by Taylor, et al. [19] and is given by

$$k = \frac{9}{40(m + 1)}\left(\frac{\rho c\dot{\epsilon}_v}{K_{IC}}\right)^2 \left(\frac{3Km}{(m + 1)\sigma_F}\right)^m \tag{4.94}$$

where σ_F is the maximum volumetric tensile stress achieved in a test to failure at a constant volumetric strain rate of $\dot{\epsilon}_v$.

If laboratory data for fracture stress versus strain rate are not available, it is possible to generate this data using an expression derived by Kipp, Grady and Chen. [24],

$$\sigma_F = \left(\frac{9\pi E K_{IC}^2}{16 N_S^2 c_s}\right)^{1/3} \dot{\epsilon}^{1/3} \tag{4.95}$$

where N_S is a shape factor (1.12 for penny shaped cracks) and c_s is the shear wave velocity of the material. Equation 4.95 has been shown to be a reasonable approximation for a number of rock types.

The damage material model uses five internal state variables:

DAMAGE	-	damage (Equation 4.82)
EVMAX	-	maximum volumetric tensile strain experienced by the material
FRAGSIZE	-	average fragment diameter (Equation 4.77)
CRKDENS	-	crack density (Equation 4.75)
EQPS	-	equivalent plastic strain

The PROP array for this material type contains the following entries:

	PROP(1)	-	Young's Modulus, E
	PROP(2)	-	Poisson's Ratio, ν
	PROP(3)	-	Yield Stress, σ_{yd}
	PROP(4)	-	m
	PROP(5)	-	k
	PROP(6)	-	Fracture Toughness, K_{IC}
*	PROP(7)	-	Bulk Modulus, K
*	PROP(8)	-	μ
*	PROP(9)	-	$m-1$
*	PROP(10)	-	COND= $\frac{5}{2}kmK_{IC}^2/(\rho c)^2$ (note: $c=\sqrt{E/\rho}$)
*	PROP(11)	-	CONA= $\frac{1}{2}(\sqrt{20}K_{IC}/\rho c)^{2/3}$

4.5 Soils and Crushable Foams Model

The soils and crushable foams model in PRONTO is a direct descendent of the model developed by Krieg [25]. Reference [25] is an unpublished Sandia National Laboratories report and is not readily available. The model was described in detail

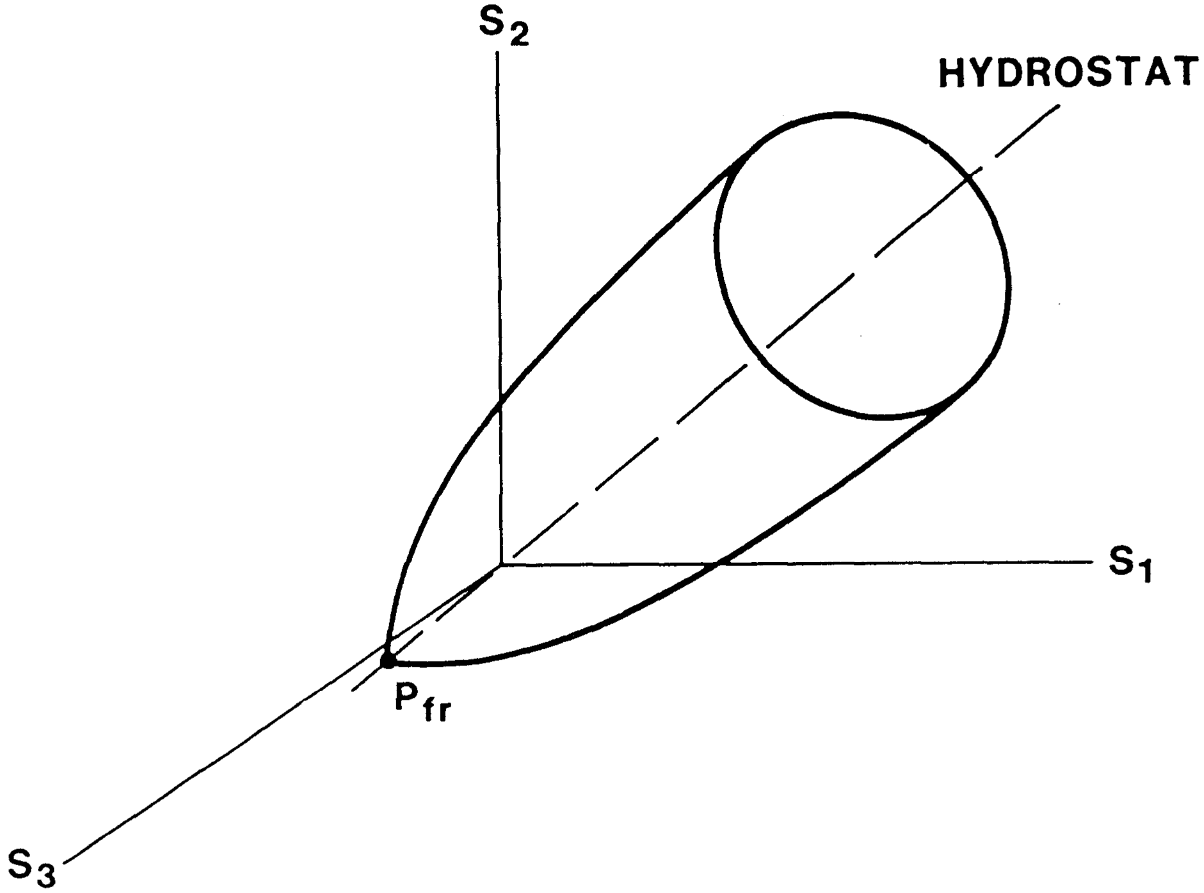


Figure 4.8. Pressure Dependent Yield Surface for the Soils and Crushable Foams Material Model

by Swenson and Taylor [26] as it was incorporated into a tensile failure model. One major difficulty with the original version of this material model which has confounded users is that the pressure dependence of the yield stress is expressed in terms of J_2, the second invariant of the stress tensor. We have reformulated the model so that the yield stress is written directly in terms of the pressure. NOTE: this means that old data must be converted.

The yield surface assumed is a surface of revolution about the hydrostat in deviatoric stress space as shown in Figure 4.8. In addition, a planar end cap on the normally open end is assumed. The yield stress is specified as a polynomial in pressure, p (positive in compression)

$$\sigma_{yd} = a_0 + a_1 P + a_2 P^2. \tag{4.96}$$

The determination of the yield stress from Equation 4.96 places severe restrictions on the admissible values of a_0, a_1, and a_2. There are three valid cases as shown in Figure 4.9. First, the user may specify a positive a_0, and a_1 and a_2 equal to zero as shown in Figure 4.9a. This gives an elastic-perfectly-plastic deviatoric response, and the yield surface is a cylinder oriented along the hydrostat in principal stress space.

Second, a conical yield surface (Figure 4.9b) is given by setting a_2 to zero and entering appropriate values of a_0 and a_1. The program checks the users input to determine whether a valid (negative) tensile fracture pressure, p_{fr}, results from the input data. The third case results when all three constants are nonzero and the program detects that a valid negative tensile failure pressure can be derived from the data. This case is shown in Figure 4.9c. A valid set of constants for the third case results in a parabola as shown in Figure 4.9c. We have drawn the descending portion of the curve with a dashed line indicating that the program does not use that portion of the curve. Instead, when the pressure exceeds p^*, the yield stress is held constant as shown at the maximum value.

The plasticity theories for the volumetric and deviatoric parts of the material response are completely uncoupled. The volumetric response is computed first. The mean pressure, p, is assumed to be positive in compression and a yield function is written for the volumetric response as

$$\phi_p = p - f_p(\epsilon_v) \tag{4.97}$$

where $f_p(\epsilon_v)$ defines the volumetric stress-strain curve for the pressure as shown in Figure 4.10. This function is defined by the user with the restriction that the slope of the function must be less than or equal to the unloading bulk modulus, K_0 , everywhere. If the user wishes the volumetric response to be purely elastic, he simply specifies no function identification (e.g., FUNCTION ID = 0). The yield function, ϕ_p, determines the motion of the end cap along the hydrostat.

The mean volumetric strain is updated as

$$\epsilon_v^{n+1} = \epsilon_v^n - \Delta t \dot{\epsilon}_v \tag{4.98}$$

where $\dot{\epsilon}_v$ is the volumetric part of the strain rate ($\dot{\epsilon}_v = \frac{1}{3} \operatorname{tr} \mathrm{d}$).

There are three possible regimes of the pressure-volumetric strain response. Tensile failure is assumed to occur if the pressure becomes smaller (more negative) than p_{fr}. The quantity ϵ_{fr} is initialized to $-p_{fr}/K_0$ by the program. If tensile failure is detected, the pressure is set to $-p_{fr}$. Remember, pressure is negative in tension! Failure by monotonic tensile loading is shown in Figure 4.11a. As long as $\epsilon_v < \epsilon_{fr}$, the pressure will remain equal to $-p_{fr}$.

If the volumetric strain exceeds ϵ_{fr}, a check is then made to see if

$$\epsilon_v < \epsilon_u \tag{4.99}$$

where ϵ_u is the most positive (compressive) volumetric strain previously experienced by the material, set initially to zero by the program. If Equation 4.99 is satisfied, the step is elastic and,

$$p^{n+1} = p^n - K_0 \Delta\epsilon_v. \tag{4.100}$$

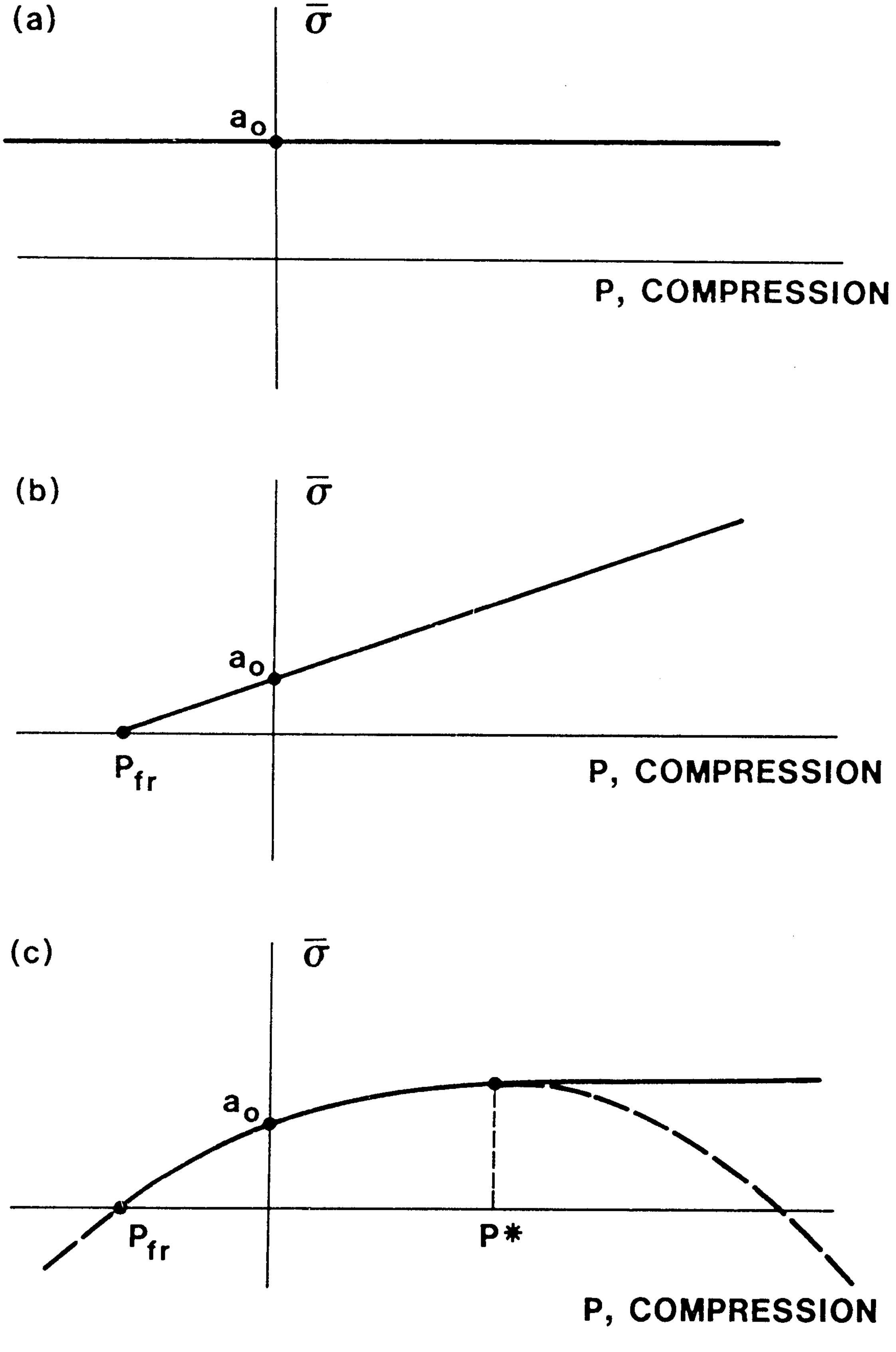


Figure 4.9. Forms of Valid Yield Surface Which can be Defined for the Soils and Crushable Foams Material Model

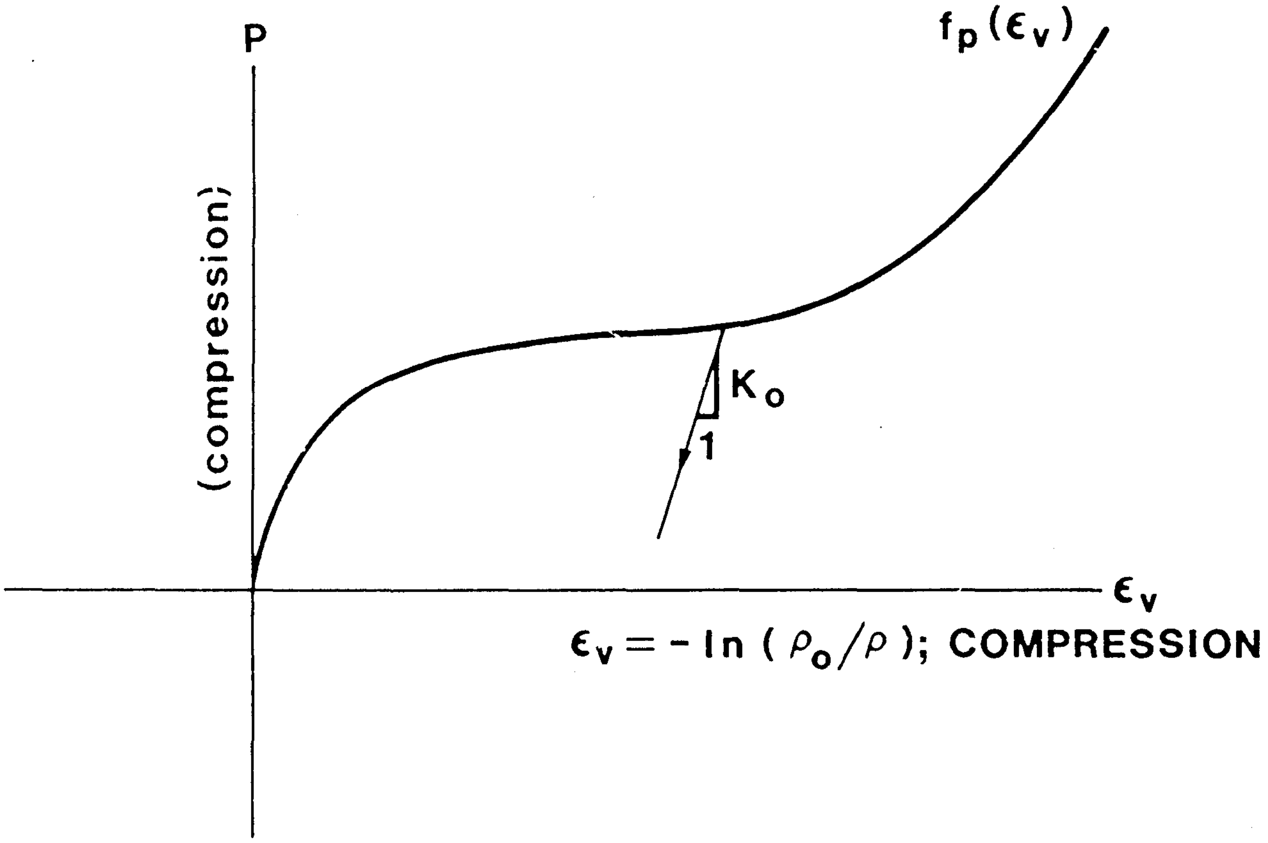


Figure 4.10. Pressure Versus Volumetric Strain Curve in Terms of a User Defined Curve. $f(\epsilon_v)$. for the Soils and Crushable Foams Material Model

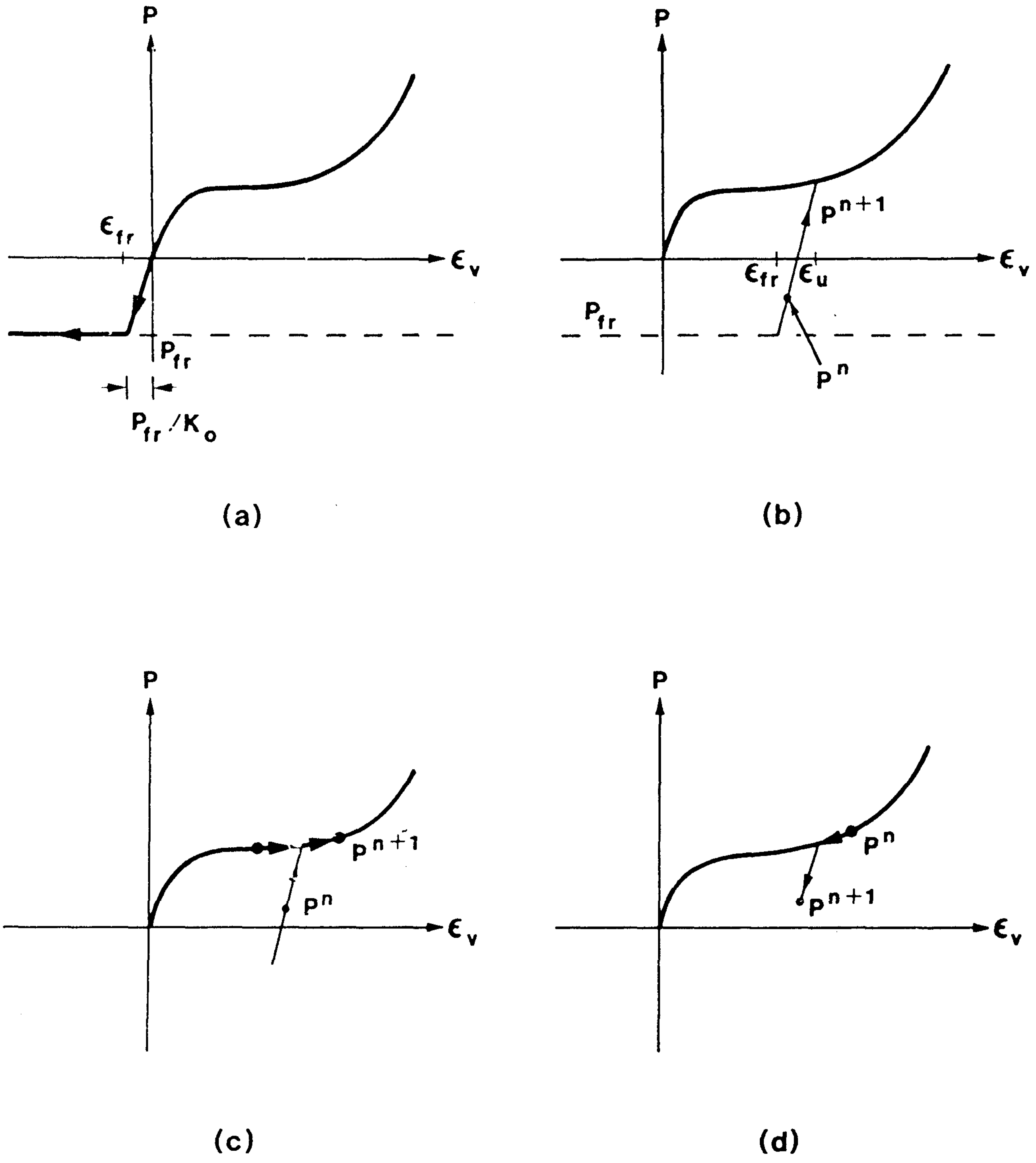


Figure 4.11. Possible Loading Cases for the Pressure Versus Volumetric Strain Response Using the Soils and Crushable Foams Material Model

This elastic response is shown in Figure 4.11b.

If Equation 4.99 is not satisfied, the volumetric response is along the curve defined by $f_p(\epsilon_v)$ and

$$p^{n+1} = f_p(\epsilon_v^{n+1}) \tag{4.101}$$

and we set

$$\epsilon_u = \epsilon_v^{n+1}. \tag{4.102}$$

This response is shown in Figure 4.11c. Note, that if Equation 4.100 is used to determine p, we also drag ϵ_{fr} along so that if we unload from the curve, $f_p(\epsilon_v)$, we will fracture at the appropriate strain level as shown in Figure 4.11d.

The deviatoric part of the response is computed next and uses a conventional plasticity theory with radial return. See Krieg and Krieg [14]. The trial elastic deviatoric stresses are computed as

$$\mathbf{S}^{tr} = \mathbf{S}_n + 2\mu\Delta t\dot{\mathbf{e}} \tag{4.103}$$

where $\dot{\mathbf{e}}$ is the deviatoric part of the strain rate. The current value of yield stress is calculated using Equation 4.96 and the vonMises effective stress, $\bar{\sigma}$, is computed as

$$\bar{\sigma} = \sqrt{\tfrac{3}{2}\mathbf{S} : \mathbf{S}} \quad . \tag{4.104}$$

The yield condition is checked to determine whether $\bar{\sigma} < \sigma_{yd}$. If this is the case, the trial stress is the correct deviatoric stress at the end of the time step, $\mathbf{S}_{n+1} = \mathbf{S}^{tr}$. If yield is exceeded, a simple radial return is performed to calculate the deviatoric stress at the end of the time step

$$\mathbf{S}_{n+1} = \frac{\sigma_{yd}}{\bar{\sigma}}\mathbf{S}^{tr}. \tag{4.105}$$

Finally, the total stress is determined by

$$\sigma_{n+1} = \mathbf{S}_{n+1} + p_{n+1}\delta \quad . \tag{4.106}$$

The Soils and Crushable Foams model uses four internal state variables:

EVMAX - maximum compressive volumetric strain experienced (always positive)

EVFRAC - current value of volumetric fracture strain (positive in compression)

EV - current value of volumetric strain (positive in compression)

NUM - integer pointing to the last increment in the pressure function where the interpolate was found

The PROP array contains the following entries for this material:

PROP(1)	-	2μ
PROP(2)	-	Bulk Modulus, K
PROP(3)	-	a_0
PROP(4)	-	a_1
PROP(5)	-	a_2
PROP(6)	-	Function ID number
PROP(7)	-	Tension Cutoff Value
* PROP(8)	-	P^*

4.6 Low Density Foams

The low density foams model presented here was developed by Neilsen, Morgan, and Krieg [27] and is based on results from experimental tests on low density, closed-cell polyurethane foams. These foams, having densities ranging from 2 to 10 pounds per cubic foot, have been proposed for use as energy absorbers in nuclear waste shipping containers. Representative responses of closed-cell polyurethane foams for various hydrostatic, uniaxial and triaxial laboratory tests indicate that the volumetric response of the foam is highly dependent on load history. This implies that typical decompositions of total foam response into an independent volumetric part and a mean stress (pressure) dependent deviatoric part are not valid for this class of foam. Many "soil and crushable foam" models, including the foam model described above in Section 4.5, use such decompositions and hence are not valid for low density closed-cell polyurethane foams. The model presented here reproduces experimental test responses more accurately for this class of foams than the model in Section 4.5.

The experimental tests on which this model is based were performed by the Civil Engineering Research Facility of the University of New Mexico with the results reported in Reference [27]. Foam samples were subjected to static, compressive stresses during these tests. In most of the tests, air was trapped in the closed cells of the foams and could not escape because the samples were jacketed with an impervious material. In this constitutive model, the total foam response is decomposed into contributions from the skeleton and from air trapped in the closed cells of the foam. The contribution of the air to the total foam response is dependent on the application. If the foam is used in a vented application where the air can escape, the contribution of the air is zero and the foam and skeleton responses are identical. If the foam is used in an application where the air cannot escape, such as a sealed shipping container, the foam pressure is considered to be the sum of the pressure carried by the skeleton and the air pressure. That is,

$$p_F = p_{sk} + p_{air} \tag{4.107}$$

where p_F and p_{sk} are the mean stress (first invariant of the stress tensor divided by three) of the foam and skeleton, respectively. The mean stresses and air pressure are assumed positive in tension. The air pressure is determined from

$$p_{air} = \frac{p_0\gamma}{1 + \gamma - \phi} \tag{4.108}$$

where γ is the engineering volume strain (first invariant of the total strains) which is positive in tension and p_0 and ϕ are model parameters. The parameter p_0 is the initial air pressure (usually atmospheric pressure of 14.7 psi), and ϕ is the ratio of the foam density to the polymer density from which the foam is produced.

Test data indicate that the skeleton response in any principal stress direction is independent of loading in any other principal stress direction. Thus, Poisson's ratio for the skeleton is equal to zero. Test data also indicate that the yield strength of the skeleton in any principal stress direction can be expressed in terms of the engineering volume strain and the second invariant of the deviatoric strains with the following relationship

$$f_i = \begin{cases} A + B(1 + C\gamma); II_{\epsilon'} > 0 \\ B(1 + C\gamma); II_{\epsilon'} = 0 \end{cases} \tag{4.109}$$

where $II_{\epsilon'}$ is the second invariant of the deviatoric strain tensor; γ is the engineering volume strain as in Equation 4.108: and A, B, and C are constants determined from fitting Equation 4.109 to the laboratory data. Constants B and C are determined from hydrostatic test data where $II_{\epsilon'}$ is zero. and A is determined from any test where the loading is deviatoric.

Numerical implementation of the model is as follows. Foam stresses and strains from the previous time increment are saved. At the beginning of the next time increment, the old skeleton stresses are computed from the old foam stresses and the old air pressure. The strain rates for the new time increment are used to determine new strain increments and trial elastic stress increments for the skeleton. These stress increments are added to the old skeleton stresses to produce new trial stresses for the skeleton. The trial skeleton stresses are then rotated to principal stress directions and compared with the yield stress determined from Equation 4.109. If yield occurs, the skeleton stresses are set to the yield stress. If yield does not occur, the trial skeleton principal stresses become the final skeleton principal stresses. The final skeleton stresses are obtained by rotating the final skeleton principal stresses back to the unrotated configuration. Then, the final foam stresses are obtained by adding the air pressure contribution for the new strain state to the new skeleton stresses.

Input parameters for the model are the constants E, p_0, ϕ, A, B, and C which are defined above. If the foam is used in an application where the air can escape, p_0 should be input as zero. Otherwise, p_0 is the atmospheric pressure at the beginning of the simulation.

The Low Density Foams model uses one internal state variable:

PAIR - internal air pressure

The PROP array contains the following entries for this material type:

PROP(1)	-	Young's Modulus, E
PROP(2)	-	A
PROP(3)	-	B
PROP(4)	-	C
PROP(5)	-	NAIR (0 = air, 1 = no air)
PROP(6)	-	p_0
PROP(7)	-	ϕ

4.7 Hydrodynamic Materials

All of the equations of state described in Chapter 5 are used by specifying a hydrodynamic material type. This material type has only volumetric or mean stress response and no deviatoric response. The pressure is calculated in the equation of state and the stress is set as

$$\sigma = -p\delta \ . \tag{4.110}$$

Note that the pressure is assumed positive in compression in the equations of state.

The hydrodynamic material requires the user to input a pressure cutoff which is positive in compression.

There are no internal state variables for this material type.

The PROP array contains only one entry for this material:

PROP(1) - Pressure cutoff

4.8 Elastic-Plastic Hydrodynamic Material

The elastic-plastic hydrodynamic material model is a combination of the elastic-plastic combined hardening model described in Section 4.2 and the purely hydrodynamic material model described in Section 4.7. In this material model, we uncouple the volumetric and deviatoric response. The volumetric response is determined using one of the equations of state defined in Chapter 5, and the deviatoric response is determined using the equations of Section 4.2.

First, we calculate the deviatoric response. This is accomplished in a manner almost identical to Section 4.2.5. We calculate a trial deviatoric stress

$$\mathbf{S}^{tr} = \mathbf{S}_n + 2\mu\dot{\mathbf{e}} \qquad (4.111)$$

where $\dot{\mathbf{e}}$ is the deviatoric part of the strain rate, $\mathbf{d}$, and 2μ is the usual Lame constant. We then proceed exactly as in Section 4.2.5 with an incremental consistency condition and determine the increment in equivalent plastic strain, update the radius of the yield surface, and update the backstress. The same radial return correction is applied to the *deviatoric* part of the stress tensor.

We then update the energy to include the increment due to the deviatoric energy contribution in Equation (5.1.9).

Next we call the appropriate equation of state to calculate the new pressure at the end of the time increment. Once this is known, we determine the total stress by

$$\sigma_{n+1} = \mathbf{S}_{n+1} - p\delta \ . \qquad (4.112)$$

The elastic-plastic hydrodynamic material model uses eight internal state variables:

EQPS - equivalent plastic strain
RADIUS - current radius of yield surface
ALPHA11 - xx component of backstress in unrotated configuration
ALPHA22 - yy component of backstress in unrotated configuration
ALPHA33 - zz component of backstress in unrotated configuration
ALPHA12 - xy component of backstress in unrotated configuration
ALPHA23 - yz component of backstress in unrotated configuration
ALPHA31 - zx component of backstress in unrotated configuration

The PROP array for this material contains the following entries:

PROP(1) - Young's Modulus, E
PROP(2) - Poisson's Ratio, ν
PROP(3) - Yield Stress, σ_{yd}
PROP(4) - Hardening Modulus, E'
PROP(5) - β
PROP(6) - pressure cutoff

- PROP(7) - 2μ
- PROP(8) - $1/(2\mu(1 - H'/3\mu))$ (Note: $H' = H/(1 - H/E)$
- PROP(9) - λ
- PROP(10) - $2\beta H'/3$
- PROP(11) - $2(1 - \beta)H'/3$

5. EQUATIONS OF STATE

The discussion of the hydrodynamic equations of state incorporated in PRONTO follows closely the development of the theory found in WONDY [28] and TOODY [29]

5.1 Introduction

The equation for conservation of energy equates the increase in internal energy per unit volume to the rate at which work is being done by the stresses and the rate at which heat is being added. In the absence of heat conduction.

$$\dot{E}_v = \rho \frac{\partial E_m}{\partial t} = (p - q)\frac{1}{\rho}\frac{\partial \rho}{\partial t} + (\sigma - p\delta) : \dot{\mathbf{e}} + \rho \dot{Q} \quad . \tag{5.1}$$

We note that in Equation 5.1. p is the pressure measured as *positive in compression*, and q is the pressure due to the bulk viscosity, from Equation 3.70. which is *negative in compression*. Also in Equation 5.1. E_v is the energy per unit volume. E_m is the energy per unit mass, and $\dot{Q}$ is the heat rate per unit mass.

The continuity equation can be written as

$$\frac{1}{\rho}\frac{\partial \rho}{\partial t} = -\operatorname{tr}\mathbf{d} = -d_{kk} \quad . \tag{5.2}$$

The deviatoric part of the strain rate is

$$\dot{\mathbf{e}} = \mathbf{d} - \frac{1}{3}\operatorname{tr}\mathbf{d}\delta \tag{5.3}$$

and the pressure is given by

$$p = -\frac{1}{3}\operatorname{tr}\sigma \quad . \tag{5.4}$$

Rewriting Equation 5.1 in terms of the pressure and the deviatoric part of the stresses. $\mathbf{S}$,

$$\dot{E}_v = (q - p)d_{kk} + \mathbf{S} : \dot{\mathbf{e}} + \rho \dot{Q} \quad . \tag{5.5}$$

An equation of state is assumed for the pressure as a function of density and energy per unit mass

$$p = f(\rho, E_m) \tag{5.6}$$

In PRONTO we use equations of state linear in internal energy of the form

$$p = f_1(\rho) + f_2(\rho)E_m \quad . \tag{5.7}$$

We find it convenient in the numerical implementation to work with energy per unit volume instead of energy per unit mass and rewrite Equation 5.7 as

$$p = f_1(\rho) + f_3(\rho)E_v \tag{5.8}$$

where we have defined a new function, $f_3(\rho) = \frac{f_2(\rho)}{\rho}$.

Assuming there are no heat sources and the strain rates are constant over the step, we can integrate Equation 5.5 to obtain the following discrete form of the energy equation:

$$E_v^{t+\Delta t} = E_v^t + \frac{\Delta t}{2}(q^t + q^{t+\Delta t} - p^t - p^{t+\Delta t})d_{kk} + \frac{\Delta t}{2}(\mathbf{S}^t + \mathbf{S}^{t+\Delta t}) : \dot{\mathbf{e}} \ . \tag{5.9}$$

Equations 5.9 and 5.8 represent two linear equations in two unknowns: $E_v^{t+\Delta t}$ and $p^{t+\Delta t}$.

Defining E_v^* by

$$E_v^* = E_v^t + \frac{\Delta t}{2}(q^t + q^{t+\Delta t} - p^t)d_{kk} + \frac{\Delta t}{2}(\mathbf{S}^t + \mathbf{S}^{t+\Delta t}) : \dot{\mathbf{e}} \ , \tag{5.10}$$

we rewrite Equation 5.9 as

$$E_v^{t+\Delta t} = E_v^* - \frac{\Delta t}{2}p^{t+\Delta t}d_{kk} \ . \tag{5.11}$$

If we have a completely hydrodynamic equation of state (see Section 4.5), there are no deviatoric terms in Equation 5.10 (i.e., $\mathbf{S}$ and $\dot{\mathbf{e}}$ are both zero). For the elastic plastic hydrodynamic material (see Section 4.9), the deviatoric and volumetric response are uncoupled. We first determine the deviatoric response and calculate the deviatoric strain energy in Equation 5.10. Then we proceed with the equation of state calculation.

The bulk viscosity pressure as defined in Equation 3.74 contains a linear and a quadratic part. Careful inspection of Equation 3.74 along with the definition of the stable time increment given by Equation 3.71 shows that the quadratic part of q is independent of the effective dilatational modulus, while the linear part is not. At the time we must calculate E_v^* in Equation 5.10, we do not yet know the effective moduli for the time step since it depends on the new pressure. To avoid the need to iterate to solve Equations 5.10 and 5.11, we do not include the energy due to the linear term in the calculation.

By substituting Equation 5.11 into 5.8, we can solve for the new pressure;

$$p^{t+\Delta t} = \frac{f_1(\rho) + f_3(\rho)E_v^*}{1 + f_3(\rho)\frac{\Delta t}{2}d_{kk}} \ . \tag{5.12}$$

After calculating the new pressure using Equation 5.12, the energy can be updated using Equation 5.11.

5.2 Mie-Gruneisen Type Equations of State

The designation Mie-Gruneisen equation of state refers to any equation of state which is linear in energy. The most general form is

$$p - p_H = \Gamma\rho(E_m - E_H) \quad . \tag{5.13}$$

where p_H and E_H are the Hugoniot pressure and specific (per unit mass) energy along some reference path and are functions of density only. The Gruneisen ratio, Γ, is also a function of density only. The Hugoniot reference pressure $p_H(\rho)$ is generally defined from fits to experimental data.

The Hugoniot specific energy is related to the Hugoniot pressure by

$$E_H = \frac{p_H\eta}{2\rho_0} \tag{5.14}$$

where

$$\eta = 1 - \frac{\rho_0}{\rho} \quad . \tag{5.15}$$

The Gruneisen ratio is usually approximated as

$$\Gamma(\rho) = \Gamma_0(1 + h_1\eta + h_2\eta^2 + \cdots) \quad . \tag{5.16}$$

Using Equation 5.14 in Equation 5.13 leads to

$$p = p_H\left[1 + \frac{\Gamma}{2}\left(\frac{\rho}{\rho_0} - 1\right)\right] + \Gamma\rho E_m \quad . \tag{5.17}$$

Equation 5.17 has the form of Equation 5.7:

$$p = f_1(\rho) + f_2(\rho)E_m \ , \tag{5.18}$$

where

$$f_1(\rho) = p_H\left(1 + \frac{\Gamma\mu}{2}\right) \quad , \tag{5.19}$$

$$f_2(\rho) = \Gamma\rho \quad , \tag{5.20}$$

and

$$\mu = \left(\frac{\rho}{\rho_0} - 1\right) = \frac{\rho}{\rho_0}\eta \quad . \tag{5.21}$$

The most common form for Equation 5.16 is to use $h_1 = -1$ and all other $h_i = 0$ which gives

$$\Gamma = \Gamma_0\frac{\rho_0}{\rho} \quad . \tag{5.22}$$

All of the Mie-Gruneisen equation of states in PRONTO use the form given by Equation 5.22.

5.2.1 Linear $U_s - U_p$ Hugoniot Form

A common fit to Hugoniot data is given by

$$p_H = \frac{\rho_0 c_0^2 \eta}{(1 - s\eta)^2} \tag{5.23}$$

where c_0 and s come from the linear shock velocity-particle velocity $U_s - U_p$ fit

$$U_s = c_0 + sU_p \ . \tag{5.24}$$

Equation 5.23 follows directly from the relations

$$p_H = U_p \rho_0 U_s \tag{5.25}$$

and

$$\eta = \frac{U_p}{U_s} \ . \tag{5.26}$$

We see that there is a limiting compression given by the denominator of Equation 5.23

$$\eta_{lim} = \frac{1}{s} \tag{5.27}$$

or

$$\rho_{lim} = \frac{s\rho_0}{s - 1} \ . \tag{5.28}$$

Also, at $\eta = -\frac{1}{s}$ there is a tensile minimum and thereafter, negative sound speeds are calculated for the material. Since Equation 5.23 is intended for use in compression, caution is advised if the model is used in response regimes where large tensions are expected.

For this form of the equation of state, we see that

$$f_1(\rho) = \frac{\rho_0 c_0^2 \eta}{(1 - s\eta)^2} \left(1 - \frac{\Gamma\mu}{2}\right) \tag{5.29}$$

and

$$f_3(\rho) = \frac{\Gamma_0 \rho_0}{\rho} = \Gamma \ . \tag{5.30}$$

5.2.2 Power Series Hugoniot Form

Another common form for the Hugoniot is to express the reference pressure, p_H, in a power series in η,

$$p_H = K_0\eta(1 + K_1\eta + K_2\eta^2 + \cdots) \ . \tag{5.31}$$

In order to match $\frac{dp_H}{d\eta}$ at $\eta = 0$ it is necessary that

$$K_0 = \rho_0 c_0^2 \tag{5.32}$$

where K_0 is the adiabatic bulk modulus at zero pressure and room temperature, and c_0 is the bulk sound speed.

For this Hugoniot form, the equation of state is defined by

$$f_1(\rho) = K_0\eta(1 + K_1\eta + K_2\eta^2)\left(1 - \frac{\Gamma\mu}{2}\right) \tag{5.33}$$

and

$$f_3(\rho) = \frac{\Gamma_0\rho_0}{\rho} = \Gamma \tag{5.34}$$

where we have restricted ourselves to using only three terms in the polynomial in Equation 5.31.

5.2.3 Ideal Gas Equation of State

The ideal gas equation of state is given by

$$p = (\gamma - 1)\rho E_m \quad . \tag{5.35}$$

where γ is a material parameter.

Hence, we see that

$$f_1(\rho) = 0 \tag{5.36}$$

and

$$f_3(\rho) = (\gamma - 1) \quad . \tag{5.37}$$

The initial sound speed in the gas, c_0, must be defined by the user. The initial pressure and specific internal energy per unit mass are

$$p_0 = \frac{\rho_0 c_0^2}{\gamma} \tag{5.38}$$

and

$$E_{m_0} = \frac{c_0^2}{\gamma(\gamma - 1)} \quad . \tag{5.39}$$

The initial pressure and energy per unit volume

$$E_0 = \rho E_{m_0} = \frac{c_0^2}{\gamma(\gamma - 1)} \tag{5.40}$$

are initialized inside the code.

5.2.4 JWL High Explosive Equation of State

The Jones-Wilkins-Lee or JWL equation of state [30] provides the pressure generated by the release of chemical energy in an explosive. In PRONTO it is implemented in a form which is usually referred to as a programmed burn. A programmed burn means that the reaction and initiation of the explosive is not determined by the shock in the material, rather the initiation time is determined by a Huygens construction using the detonation wave speed and the distance of the material point from the detonation point(s).

The JWL equation of state is generally written as

$$p = A\left(1-\frac{\omega\rho}{R_1\rho_0}\right)e^{\left(-R_1\frac{\rho_0}{\rho}\right)} + B\left(1-\frac{\omega\rho}{R_2\rho_0}\right)e^{\left(-R_2\frac{\rho_0}{\rho}\right)} + \frac{\omega\rho^2}{\rho_0}E_{m_0} \tag{5.41}$$

where A, B, R_1, R_2, ω, and E_{m_0} are material constants. Note that Equation 5.41 is written in terms of energy per mass which is the usual form found in the literature. Again, we chose to write our equations of state in terms of energy per unit volume which results in

$$f_1(\rho) = A\left(1-\frac{\omega\rho}{R_1\rho_0}\right)e^{\left(-R_1\frac{\rho_0}{\rho}\right)} + B\left(1-\frac{\omega\rho}{R_2\rho_0}\right)e^{\left(-R_2\frac{\rho_0}{\rho}\right)} \tag{5.42}$$

and

$$f_3(\rho) = \frac{\omega\rho}{\rho_0} \; . \tag{5.43}$$

The programmed burn requires the initial calculation of the arrival of the detonation wave at a material point. If there is only one detonation point denoted by x_d, and if the location of the material point is denoted by x_n, then the detonation time is determined by

$$t_d = \frac{\mid x_d - x_n \mid}{c_d} \tag{5.44}$$

where c_d is the detonation wave speed (a material property supplied by the user) and the symbol $\mid \cdot \mid$ indicates the euclidean norm of a vector. Clearly, if there are multiple detonation points, then Equation 5.44 must be applied for each material point for each detonation point and the arrival time is the minimum.

In order to spread the burn wave over several elements, a burn fraction F is computed as

$$F = \min\left[1, \frac{(t-t_d)c_d}{B_s l}\right] \tag{5.45}$$

where B_s is a constant which controls the width of the burn wave (defaulted to 2.5 in the code) and l is the characteristic length of the element which is calculated internally in the code as the square root of the area of the element. If the time t is less than t_d , the pressure is zero in the explosive. Otherwise, the pressure is given by

$$p = F\left[f_1(\rho) + f_2(\rho)E_v\right] \quad . \tag{5.46}$$

When $t < t_d$ the detonation wave speed is used as the sound speed in the material. After the detonation wave has arrived, the sound speed is calculated internally from the hypoelastic stress rates and strain rates just as for all other materials.

This form of the equation of state requires that the internal energy per unit volume be initialized to account for the chemical energy in the explosive. Parameters for a wide variety of explosives have been tabulated by Dobratz [31].

6. CONTACT SURFACES

PRONTO currently supports two types of contact surface boundary conditions: a deformable surface against a rigid plane, and two distinct deformable surfaces against each other. The first option requires a far simpler procedure since the constraints on each node are completely uncoupled.

Contact is treated as a kinematic constraint by PRONTO. This means that the final product of the contact algorithm is to modify the accelerations of the nodes along these surfaces such that the kinematic constraints are satisfied.

PRONTO supports friction for both contact surface options. Either a simple Coulomb friction model or a velocity-dependent friction model may be selected.

6.1 Deformable-to-Rigid Surface Contact

The rigid surface option in PRONTO imposes the kinematic constraints of an unyielding plane on a user-specified surface of the deformable body. The plane is defined by a point $\bar{\mathbf{x}}$ and the outward unit normal $\mathbf{n}$. The deformable surface can be treated as simply a set of unique nodes. The primary kinematic condition is that the deformable nodes may not penetrate the rigid plane. In addition, the motion of deformable nodes along the plane may be restricted subject to a velocity-dependent friction law.

6.1.1 Normal Constraint

We begin by integrating the motion of the deformable nodes without regard to the kinematic constraints required by the rigid surface. For each node, we calculate a predicted kinematic state as follows:

$$\hat{\mathbf{a}} = \frac{\mathbf{f}}{m} \tag{6.1}$$

$$\hat{\mathbf{v}} = \mathbf{v} + \Delta t\, \hat{\mathbf{a}} \tag{6.2}$$

$$\hat{\mathbf{x}} = \mathbf{x} + \Delta t\, \hat{\mathbf{v}} \tag{6.3}$$

In the above equations, $\mathbf{f}$ is the residual force vector (sum of external forces minus sum of internal forces), m is the nodal mass, $\mathbf{v}$ is the current velocity, $\mathbf{x}$ is the current position, and Δt is the time increment. The predicted kinematic quantities are denoted by a superposed hat. The predicted velocities are temporarily stored in the residual force array.

We now calculate the depth of penetration δ of each node into the plane as given below. This depth is zero for nodes which are not in contact.

$$\delta = \max(\mathbf{n}\cdot(\bar{\mathbf{x}} - \hat{\mathbf{x}}), 0) \tag{6.4}$$

The magnitude of the force which must be applied to enforce the kinematic constraint, i.e., which will cancel the penetration, is given by

$$f_n = \frac{\delta m}{\Delta t^2} . \tag{6.5}$$

This force must be applied in the direction of $\mathbf{n}$. Applying this correction to Equation 6.1 and eliminating the nodal mass, we can express the new acceleration in the absence of friction as

$$a_n = \frac{\delta}{\Delta t^2} ; \tag{6.6}$$

$$\mathbf{a} = \hat{\mathbf{a}} + a_n \mathbf{n} . \tag{6.7}$$

6.1.2 Friction

Friction resists tangential motion of deformable nodes contacting the rigid plane. The predicted tangential velocity of a node is orthogonal to the outward normal and, therefore, is expressed by

$$\mathbf{v}_s = \hat{\mathbf{v}} - (\mathbf{n} \cdot \hat{\mathbf{v}})\mathbf{n} . \tag{6.8}$$

The above velocity is decomposed into a magnitude and unit direction vector as follows:

$$v_s = \sqrt{\mathbf{v}_s \cdot \mathbf{v}_s} . \tag{6.9}$$

$$\mathbf{s} = \frac{\mathbf{v}_s}{v_s} \tag{6.10}$$

The force which must be applied to cancel the tangential velocity of a node is then given by

$$f_s = -\frac{m v_s}{\Delta t} . \tag{6.11}$$

where the minus sign above reflects that this force would be applied in the direction of s, but opposing the motion.

PRONTO currently supports three options for friction: no friction, Coulomb friction with a constant coefficient of friction, or the velocity-dependent friction law found in HONDO II [32]. The coefficient of friction can be expressed by

$$\mu = \mu_\infty + (\mu_0 - \mu_\infty)e^{-\gamma v_s} \tag{6.12}$$

where μ_0 and μ_∞ are the low- and high-velocity friction coefficients, respectively, and γ is a decay constant. Clearly, if γ equals zero, the coefficient of friction is the constant μ_0. Futhermore, if μ_0 is also equal to zero, the surface will be frictionless.

The magnitude of the tangential force exerted by the plane on a node cannot exceed the maximum friction force. This constraint is expressed as

$$f_f = \frac{f_s}{|f_s|} \min(\mu f_n, |f_s|) . \tag{6.13}$$

Substituting Equations 6.5, 6.6, and 6.11 into the above, then eliminating the nodal mass yields

$$a_s = -\frac{v_s}{|v_s|} \min(\mu a_n, \frac{|v_s|}{\Delta t}) . \tag{6.14}$$

The total acceleration of the node is then given by

$$\mathbf{a} = \hat{\mathbf{a}} + a_n \mathbf{n} + a_s \mathbf{s} . \tag{6.15}$$

6.2 Deformable-to-Deformable Surface Contact

The fundamental condition which must be satisfied between two contact surfaces is that one surface may not penetrate into the other. The algorithmic challenge is to find the set of nodal forces which will maintain kinematic compliance. The classic difficulty of contact algorithms is that elaborate and exhaustive schemes to maintain strict compliance are prohibitively expensive to implement and to execute. Furthermore, because the surface is discretized for the finite element method, it is virtually impossible to pose an algorithm which always yields unique and meaningful results.

The contact algorithm in PRONTO is designed to handle very large deformations and high impact velocities. The goal of the contact algorithm is that it function properly to the capability limits of a Lagrangian mesh. The sample problems in Chapter 9 illustrate a fairly representative, but by no means exhaustive, range of applicability of the PRONTO contact surfaces.

PRONTO uses a partitioned kinematic approach to contact. The partitioning can be adjusted to give a strict master-slave treatment or to balance the master-slave relationship between the two surfaces. In any case, the constraint forces conserve momentum.

The contact algorithm is performed in two passes; first with one surface as the master, then with the other surface as the master. One of these passes will be skipped if a strict master-slave treatment is requested. The following sections describe just one such pass.

6.2.1 Surface Topology

A surface in PRONTO must be continuous and simply-connected. It may be either an open surface with a continuous, simoly-connected perimeter or a closed surface, such as the surface of a sphere. The 'inside' of the surface to PRONTO is where the material lies; it is assumed that there is no material on the 'outside' of the surface. In order for a surface node to form a valid connection, all adjoining faces must have material on the same side.

PRONTO carefully checks the topology of contact surfaces during initialization. It will print an appropriate error message(s) if the surface is either multiply connected or discontinuous. If the surface is found to be valid, PRONTO will build the data structures which describe the topology of the surface as described below.

The *node list* data structure, `NODJCT(NNODES)`, uniquely contains the nodes which appear on a given surface. Most nodal data within PRONTO's contact algorithm are referenced by surface node index, rather than global node index. From this perspective, the node list data structure contains the global node index of each surface node.

The *facial node map* data structure, `JCTFAC(4,NFACES)`, contains the surface node indices of each face on a given surface. PRONTO uses the right-hand rule to define the direction of the outward normal. In other words, the facial nodes circulate counter-clockwise while looking at the outside of-the face.

The *nodal face map* data structure, `KNRJCT(MAXFAC,NNODES)`, contains the index of each face connected to each surface node. By convention, the faces circulate clockwise around the node while looking at the outside of the surface. This data structure is made rectangular, based on the maximum number of faces connected to any node on the surface, for efficiency. The *face counter* data structure, `KNTFAC(NNODES)`, contains the number of faces actually connected to each surface node. A row of the nodal face map about node I, which is contained within `KNRJCT(1:KNTFAC(I),NNODES)`, represents a closed loop (circular list) of faces for an interior node, or an open loop (straight list) of faces for a perimeter node.

6.2.2 Surface Geometry

PRONTO recalculates the geometry of all contact surfaces at each time step. A predicted configuration is computed by integrating the motion without regard to the kinematic constraints required by the contact surfaces. For each node we calculate:

$$\hat{\mathbf{a}} = \frac{\mathbf{f}}{m} \tag{6.16}$$

$$\hat{\mathbf{v}} = \mathbf{v} + \Delta t\, \hat{\mathbf{a}} \tag{6.17}$$

$$\hat{\mathbf{x}} = \mathbf{x} + \Delta t\, \hat{\mathbf{v}} \tag{6.18}$$

In the above equations, $\mathbf{f}$ is the residual force vector (sum of external forces minus sum of internal forces), m is the nodal mass, $\mathbf{v}$ is the current velocity, $\mathbf{x}$ is the current position, and Δt is the time increment. The predicted kinematic quantities are denoted by a superposed hat. The predicted accelerations are never stored in a global array.

PRONTO uses the *average surface normal* to resolve contact at corners. This vector quantity is derived for each surface node from the mean face normals connected to that node. First, the mean face normal of each face is calculated and normalized as follows.

$$\mathbf{N}_I = (\mathbf{x}_2 + \mathbf{x}_3 - \mathbf{x}_1 - \mathbf{x}_4) \times (\mathbf{x}_2 + \mathbf{x}_4 - \mathbf{x}_1 - \mathbf{x}_3) \tag{6.19}$$

$$\mathbf{n}_I = \frac{\mathbf{N}_I}{\sqrt{\mathbf{N}_I \cdot \mathbf{N}_I}} \tag{6.20}$$

The average surface normal for a given surface node is then calculated by summing the unit face normals of each face connected to that node. Since PRONTO admits highly warped (nonplanar) faces in the contact algorithm, it is not useful to retain the face normals.

6.2.3 Surface Tracking

Surface tracking is the process of matching points along one surface to points along its mating surface. PRONTO's tracking algorithm hinges on finding the *nearest point* on the master surface to each slave node. This is a nonunique point which lies within one or more master faces. The *nearest face* is the master face containing the nearest point on the surface. Once we have determined the nearest face, all that remains is to determine if the slave node is contacting that face or one of its neighbors. In the next section (6.2.4), we describe how we decide whether the slave node is in contact and which master face it is contacting.

The tracking algorithm truly governs the cost/benefit of the contact surface capability in PRONTO. The amount of geometric detail that the tracking algorithm can resolve determines the range of applicability of the contact surfaces. On the other hand, exhaustive checks of every slave node against every master face during every time step will always be prohibitively expensive, and generally unwarranted. The tracking algorithm, therefore, is an area where compromises must be made, but where cleverness will pay off.

PRONTO tracks the nearest master node for each slave node. It is important to understand that in this context the 'nearness' of a node is measured in terms of the surficial (along the surface) distance, rather than the spatial (straight line) distance. The *nearest node*, therefore, is defined as the nearest facial node to the nearest point within the nearest face. The preciseness of this definition is crucial to resolving sharp corners, as illustrated in Figure 6.1.

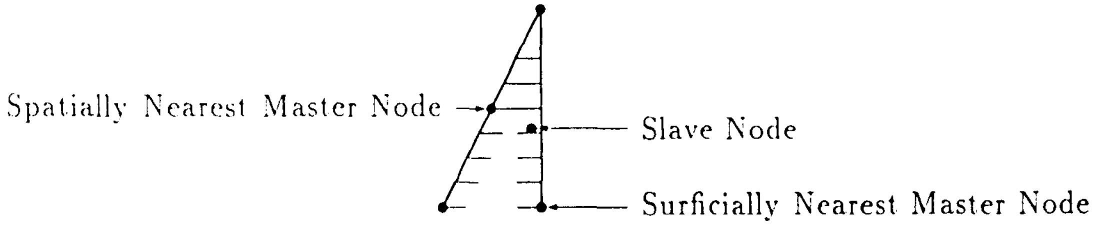

Spatially Nearest Master Node
Slave Node
Surficially Nearest Master Node

Figure 6.1. Surficial Versus Spatial Distance

To streamline the tracking algorithm in PRONTO, we assume that the nearest master node to a given slave node at one time step will be in the vicinity of the nearest master node at the next time step. This assumption allows us to update the tracking scheme by simply searching for a local minimum in the vicinity of the previously nearest master node. Thus, at each time step, we start at the previously nearest node and search via the nodal face map for the nearest point and face among its connected faces. We then find the new nearest master node. If the nearest node changes, we update and continue the search process, until a stable nearest node is attained.

We initialize the tracking scheme for each slave node S by simply finding the spatially nearest master node I, which has the minimum distance d_I defined by

$$d_I^2 = (\mathbf{x}_I - \mathbf{x}_S) \cdot (\mathbf{x}_I - \mathbf{x}_S) \ . \tag{6.21}$$

The tracking algorithm could fail if the local minimum scheme does not find the global minimum. This failure is extremely unlikely to occur (in fact, it has never been observed), because incremental geometric changes over a time step are forced to be small by the explicit integration scheme. The situation could be remedied, in any case, by simply restarting the calculation (Appendix A, command 4 and 5) at an appropriate time. PRONTO reinitializes the tracking data when it reads a restart file, which forces an exhaustive search for the nearest master node via Equation 6.21.

In addition to the pathological cases described above, the tracking algorithm in PRONTO does not yet support a surface contacting itself. This capability would be useful for buckling shells which fold upon themselves. The problem in this instance is that the tracking scheme will always find that each node is contacting itself. Presently, the only way to handle this situation is to divide the surface at the crease points. The best approach is to run the calculation with fairly frequent restart dumps (Appendix A, command 5), identify a restart state which occurs after the surface has buckled, but before contact, then restart from this state (Appendix A, *command* 4) with the proper contact surfaces inserted. The greatest difficulty with this technique is that it requires manipulation of the GENESIS mesh file [2].

The success of the tracking scheme depends most heavily on determining the nearest point on an element face to a given slave node. This amounts to a nonlinear constrained minimization problem. The vector from a slave node S to a point on an element face (represented by face nodes 1 → 4 is expressed in terms of the isoparametric face coordinates as follows.

$$\mathbf{d} = \mathbf{t} + \xi\mathbf{u} + \eta\mathbf{v} + \xi\eta\mathbf{w} \tag{6.22}$$

where

$$\mathbf{t} = \tfrac{1}{4}(+\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3 + \mathbf{x}_4) - \mathbf{x}_S \tag{6.23}$$

$$\mathbf{u} = \tfrac{1}{2}(-\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3 - \mathbf{x}_4) \tag{6.24}$$

$$\mathbf{v} = \tfrac{1}{2}(-\mathbf{x}_1 - \mathbf{x}_2 + \mathbf{x}_3 + \mathbf{x}_4) \tag{6.25}$$

$$\mathbf{w} = (+\mathbf{x}_1 - \mathbf{x}_2 + \mathbf{x}_3 - \mathbf{x}_4) \tag{6.26}$$

We seek ξ and η which minimize $\mathbf{d} \cdot \mathbf{d}$ subject to the following constraints.

$$-\tfrac{1}{2} \leq \xi \leq +\tfrac{1}{2} \tag{6.27}$$

$$-\tfrac{1}{2} \leq \eta \leq +\tfrac{1}{2} \tag{6.28}$$

Since Equation 6.22 is bilinear, the inner product of $\mathbf{d}$ is fourth-order. Since the inner product is also positive definite, it cannot have a maximum. Therefore, the general case has two minima between a saddle point. The most rapidly converging technique for nonlinear minimization is Newton's method. However, Newton's method has two problems in regard to our application. First, it requires a good initial estimate to avoid the saddle point. Second, it does not enforce the constraints.

The technique implemented in PRONTO for finding the nearest point generates a good initial estimate which properly accounts for the constraints. Newton's method is then applied to improve this estimate. PRONTO starts by finding the constrained minimum, i.e., the nearest point on the perimeter. The nearest point on each edge is given by the following equations, respectively.

$$\hat{\xi}_1 = \max\left(-\tfrac{1}{2}, \min\left(+\tfrac{1}{2}, -\frac{(\mathbf{t} - \frac{1}{2}\mathbf{v}) \cdot (\mathbf{u} - \frac{1}{2}\mathbf{w})}{(\mathbf{u} - \frac{1}{2}\mathbf{w}) \cdot (\mathbf{u} - \frac{1}{2}\mathbf{w})}\right)\right) \tag{6.29}$$

$$\hat{\eta}_2 = \max\left(-\tfrac{1}{2}, \min\left(+\tfrac{1}{2}, -\frac{(\mathbf{t} + \frac{1}{2}\mathbf{u}) \cdot (\mathbf{v} + \frac{1}{2}\mathbf{w})}{(\mathbf{v} + \frac{1}{2}\mathbf{w}) \cdot (\mathbf{v} + \frac{1}{2}\mathbf{w})}\right)\right) \tag{6.30}$$

$$\hat{\xi}_3 = \max\left(-\tfrac{1}{2}, \min\left(+\tfrac{1}{2}, -\frac{(\mathbf{t}+\frac{1}{2}\mathbf{v})\cdot(\mathbf{u}+\frac{1}{2}\mathbf{w})}{(\mathbf{u}+\frac{1}{2}\mathbf{w})\cdot(\mathbf{u}+\frac{1}{2}\mathbf{w})}\right)\right) \tag{6.31}$$

$$\hat{\eta}_4 = \max\left(-\tfrac{1}{2}, \min\left(+\tfrac{1}{2}, -\frac{(\mathbf{t}-\frac{1}{2}\mathbf{u})\cdot(\mathbf{v}-\frac{1}{2}\mathbf{w})}{(\mathbf{v}-\frac{1}{2}\mathbf{w})\cdot(\mathbf{v}-\frac{1}{2}\mathbf{w})}\right)\right) \tag{6.32}$$

The *nearest perimeter point* is the nearest of the above points, whose respective distances are given as follows.

$$\min_{\eta=-\frac{1}{2}} \mathbf{d}\cdot\mathbf{d} = (\mathbf{t}-\tfrac{1}{2}\mathbf{v})\cdot(\mathbf{t}-\tfrac{1}{2}\mathbf{v}) + \hat{\xi}_1(\mathbf{t}-\tfrac{1}{2}\mathbf{v})\cdot(\mathbf{u}-\tfrac{1}{2}\mathbf{w}) \tag{6.33}$$

$$\min_{\xi=+\frac{1}{2}} \mathbf{d}\cdot\mathbf{d} = (\mathbf{t}+\tfrac{1}{2}\mathbf{u})\cdot(\mathbf{t}+\tfrac{1}{2}\mathbf{u}) + \hat{\eta}_2(\mathbf{t}+\tfrac{1}{2}\mathbf{u})\cdot(\mathbf{v}+\tfrac{1}{2}\mathbf{w}) \tag{6.34}$$

$$\min_{\eta=+\frac{1}{2}} \mathbf{d}\cdot\mathbf{d} = (\mathbf{t}+\tfrac{1}{2}\mathbf{v})\cdot(\mathbf{t}+\tfrac{1}{2}\mathbf{v}) + \hat{\xi}_3(\mathbf{t}+\tfrac{1}{2}\mathbf{v})\cdot(\mathbf{u}+\tfrac{1}{2}\mathbf{w}) \tag{6.35}$$

$$\min_{\xi=-\frac{1}{2}} \mathbf{d}\cdot\mathbf{d} = (\mathbf{t}-\tfrac{1}{2}\mathbf{u})\cdot(\mathbf{t}-\tfrac{1}{2}\mathbf{u}) + \hat{\eta}_4(\mathbf{t}-\tfrac{1}{2}\mathbf{u})\cdot(\mathbf{v}-\tfrac{1}{2}\mathbf{w}) \tag{6.36}$$

The next step is to estimate an unconstrained minimum by searching for a nearer interior point. To accomplish this task, PRONTO uses edge triangles; these are the four triangles formed by an edge and the face center. The edge triangles properly approximate all of the significant features of the warping of a bilinear element face, but since they are planes, it is straightforward to find their respective nearest points. First, we need the nearest point to each of the major chords, which are given below.

$$\hat{\eta}_0 = -\frac{\mathbf{t}\cdot\mathbf{v}}{\mathbf{v}\cdot\mathbf{v}} \tag{6.37}$$

$$\hat{\xi}_0 = -\frac{\mathbf{t}\cdot\mathbf{u}}{\mathbf{u}\cdot\mathbf{u}} \tag{6.38}$$

The nearest point to each edge triangle can be determined from Equations 6.29–6.32 and Equations 6.37–6.38, subject to both the external constraints of the edge and the internal constraints of the triangle. The results are as follows.

$$\eta_1 = \max\left(-\tfrac{1}{2}, \min\left(\frac{|\hat{\xi}_1 - \frac{1}{2}\alpha_1|}{\pm\alpha_1 - 1}, \frac{\hat{\eta}_0 - \theta_1\hat{\xi}_1 + \frac{1}{2}\alpha_1\theta_1}{1-\alpha_1\theta_1}\right)\right) \tag{6.39}$$

$$\xi_2 = \min\left(+\tfrac{1}{2}, \max\left(\frac{|\hat{\eta}_2 + \frac{1}{2}\alpha_2|}{\pm\alpha_2 + 1}, \frac{\hat{\xi}_0 - \theta_2\hat{\eta}_2 - \frac{1}{2}\alpha_2\theta_2}{1-\alpha_2\theta_2}\right)\right) \tag{6.40}$$

$$\eta_3 = \min\left(+\tfrac{1}{2}, \max\left(\frac{|\hat{\xi}_3 + \frac{1}{2}\alpha_3|}{\pm\alpha_3 + 1}, \frac{\hat{\eta}_0 - \theta_3\hat{\xi}_3 - \frac{1}{2}\alpha_3\theta_3}{1-\alpha_3\theta_3}\right)\right) \tag{6.41}$$

$$\xi_4 = \max\left(-\tfrac{1}{2}, \min\left(\frac{|\hat{\eta}_4 - \frac{1}{2}\alpha_4|}{\pm\alpha_4 - 1}, \frac{\hat{\xi}_0 - \theta_4\hat{\eta}_4 + \frac{1}{2}\alpha_4\theta_4}{1-\alpha_4\theta_4}\right)\right) \tag{6.42}$$

$$\xi_1 = \hat{\xi}_1 - \alpha_1(\eta + \tfrac{1}{2}) \tag{6.43}$$

$$\eta_2 = \hat{\eta}_2 - \alpha_2(\xi + \tfrac{1}{2}) \tag{6.44}$$

$$\xi_3 = \hat{\xi}_3 - \alpha_3(\eta - \tfrac{1}{2}) \tag{6.45}$$

$$\eta_4 = \hat{\eta}_4 - \alpha_4(\xi - \tfrac{1}{2}) \tag{6.46}$$

where the indeterminate sign in the denominator of the internal constraint in Equations 6.39–6.42 takes the sign of the quantity in the numerator. The other quantities introduced above are evaluated as follows.

$$\alpha_1 = \frac{(\mathbf{u} - \frac{1}{2}\mathbf{w}) \cdot \mathbf{v}}{(\mathbf{u} - \frac{1}{2}\mathbf{w}) \cdot (\mathbf{u} - \frac{1}{2}\mathbf{w})} \tag{6.47}$$

$$\alpha_2 = \frac{(\mathbf{v} + \frac{1}{2}\mathbf{w}) \cdot \mathbf{u}}{(\mathbf{v} + \frac{1}{2}\mathbf{w}) \cdot (\mathbf{v} + \frac{1}{2}\mathbf{w})} \tag{6.48}$$

$$\alpha_3 = \frac{(\mathbf{u} + \frac{1}{2}\mathbf{w}) \cdot \mathbf{v}}{(\mathbf{u} + \frac{1}{2}\mathbf{w}) \cdot (\mathbf{u} + \frac{1}{2}\mathbf{w})} \tag{6.49}$$

$$\alpha_4 = \frac{(\mathbf{v} - \frac{1}{2}\mathbf{w}) \cdot \mathbf{u}}{(\mathbf{v} - \frac{1}{2}\mathbf{w}) \cdot (\mathbf{v} - \frac{1}{2}\mathbf{w})} \tag{6.50}$$

$$\theta_1 = \frac{(\mathbf{u} - \frac{1}{2}\mathbf{w}) \cdot \mathbf{v}}{\mathbf{v} \cdot \mathbf{v}} \tag{6.51}$$

$$\theta_2 = \frac{(\mathbf{v} + \frac{1}{2}\mathbf{w}) \cdot \mathbf{u}}{\mathbf{u} \cdot \mathbf{u}} \tag{6.52}$$

$$\theta_3 = \frac{(\mathbf{u} + \frac{1}{2}\mathbf{w}) \cdot \mathbf{v}}{\mathbf{v} \cdot \mathbf{v}} \tag{6.53}$$

$$\theta_4 = \frac{(\mathbf{v} - \frac{1}{2}\mathbf{w}) \cdot \mathbf{u}}{\mathbf{u} \cdot \mathbf{u}} \tag{6.54}$$

The final step in PRONTO's nearest point location algorithm is to improve the edge triangle estimate via Newton's method. The solution estimate is iteratively improved by solving the following system of equations:

$$\begin{bmatrix} (\mathbf{u} + \eta\mathbf{w}) \cdot (\mathbf{u} + \eta\mathbf{w}) & (\mathbf{u} + \eta\mathbf{w}) \cdot (\mathbf{v} + \xi\mathbf{w}) + \mathbf{d} \cdot \mathbf{w} \\ (\mathbf{u} + \eta\mathbf{w}) \cdot (\mathbf{v} + \xi\mathbf{w}) + \mathbf{d} \cdot \mathbf{w} & (\mathbf{v} + \xi\mathbf{w}) \cdot (\mathbf{v} + \xi\mathbf{w}) \end{bmatrix} \begin{bmatrix} \Delta\xi \\ \Delta\eta \end{bmatrix} = \begin{bmatrix} -\mathbf{d} \cdot (\mathbf{u} + \eta\mathbf{w}) \\ -\mathbf{d} \cdot (\mathbf{v} + \xi\mathbf{w}) \end{bmatrix} \tag{6.55}$$

Three Newton iterations are performed in PRONTO. This assures approximately three digit precision in determining the nearest point. Note that if the external constraint in Equations 6.39–6.42 has to be enforced, no unconstrained minimum exists; therefore, the Newton iterations are ignored.

6.2.4 Determination of Contact

PRONTO's approach to determining contact is based on the premise that a slave node usually contacts the nearest face as defined in the previous section. PRONTO requires two conditions for contact: the slave node must be *penetrating* and *within* a master face. The node is penetrating if and only if it projects to the inside of the face at the nearest point. The condition is expressed as follows:

$$\mathbf{d} \cdot (\mathbf{u} + \eta \mathbf{w}) \times (\mathbf{v} + \xi \mathbf{w}) < 0 \tag{6.56}$$

The node is within the face if and only if it projects normal to the face at the nearest point. This strictly requires that the right-hand side of Equation 6.55 vanish. In practice, we determine if the node projects within ξ and η, respectively, as follows:

$$\left| \xi + \frac{\mathbf{d} \cdot (\mathbf{u} + \eta \mathbf{w})}{(\mathbf{u} + \eta \mathbf{w}) \cdot (\mathbf{u} + \eta \mathbf{w})} \right| < \tfrac{1}{2} + \epsilon \tag{6.57}$$

$$\left| \eta + \frac{\mathbf{d} \cdot (\mathbf{v} + \xi \mathbf{w})}{(\mathbf{v} + \xi \mathbf{w}) \cdot (\mathbf{v} + \xi \mathbf{w})} \right| < \tfrac{1}{2} + \epsilon \tag{6.58}$$

where ϵ is a small dimensionless number; we use $\epsilon = 0.001$.

The ideal condition for determining contact is that the slave node is penetrating and projects within exactly one master face. Unfortunately, this definition leads to several ambiguous cases because the surface normal is not continuous. One must impose further conditions in order to resolve these ambiguities.

Since the surface normal is not continuous across element faces, the surface generally forms a crease at an edge. When the angle between two faces becomes significant, we refer to the edge as a *corner*. There are two general cases, inside and outside corners, which pose distinctly different problems for the contact algorithm.

When two adjacent faces form an inside corner, there is crevice where a slave node could be penetrating the surface, but not project normal to either face. This case is illustrated in Figure 6.2. PRONTO's solution to this ambiguity is to relax the applicable Condition 6.57 or 6.58 for the nearest face if the slave node is penetrating the adjacent face. This situation does not apply to a free perimeter edge since there is no adjacent face.

In the case of an outside corner, there is an overlap where a slave node could be contacting either face, as illustrated in Figure 6.3. In this instance, the average surface normal defined in Section 6.2.2 is used to resolve the ambiguity. If the average surface normal of the slave node is more strongly opposed to the adjacent face than to the nearest face, the adjacent face has priority for contact. The adjacent face, however,

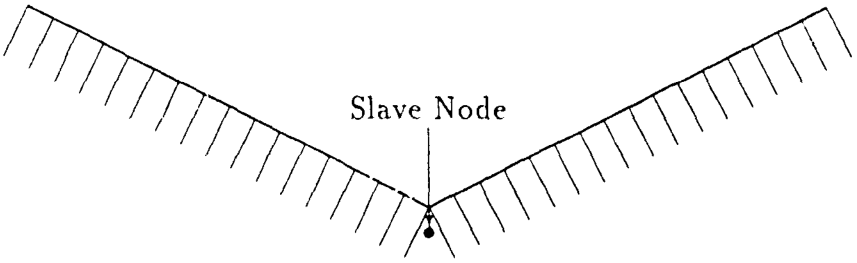


Figure 6.2. Inside Corner

must further satisfy Conditions 6.56–6.58 to be in contact. To perform the opposition tests, we must first normalize the projection vector to each face as follows:

$$\delta = \sqrt{\mathbf{d} \cdot \mathbf{d}} \tag{6.59}$$

$$\mathbf{m} = \frac{\mathbf{d}}{\delta} \tag{6.60}$$

We then determine if the average surface normal of the slave node, $\mathbf{n}_S$, is more opposed to the adjacent face normal, $\mathbf{m}_A$, than the nearest face normal, $\mathbf{m}_N$, as follows:

$$\mathbf{n}_S \cdot \mathbf{m}_A < \mathbf{n}_S \cdot \mathbf{m}_N \tag{6.61}$$

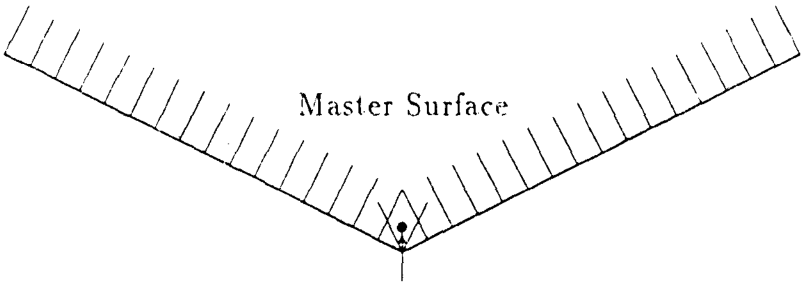


Figure 6.3. Outside Corner

The logic that PRONTO follows to determine contact once it has identified the nearest face is summarized in Figure 6.4. In this figure, *precedent* and *antecedent* refer to faces adjacent to the nearest face within the nodal face map (Section 6.2.1) about the nearest node. Note that there are only three basic questions. This logic is so simple that it can be vectorized efficiently.

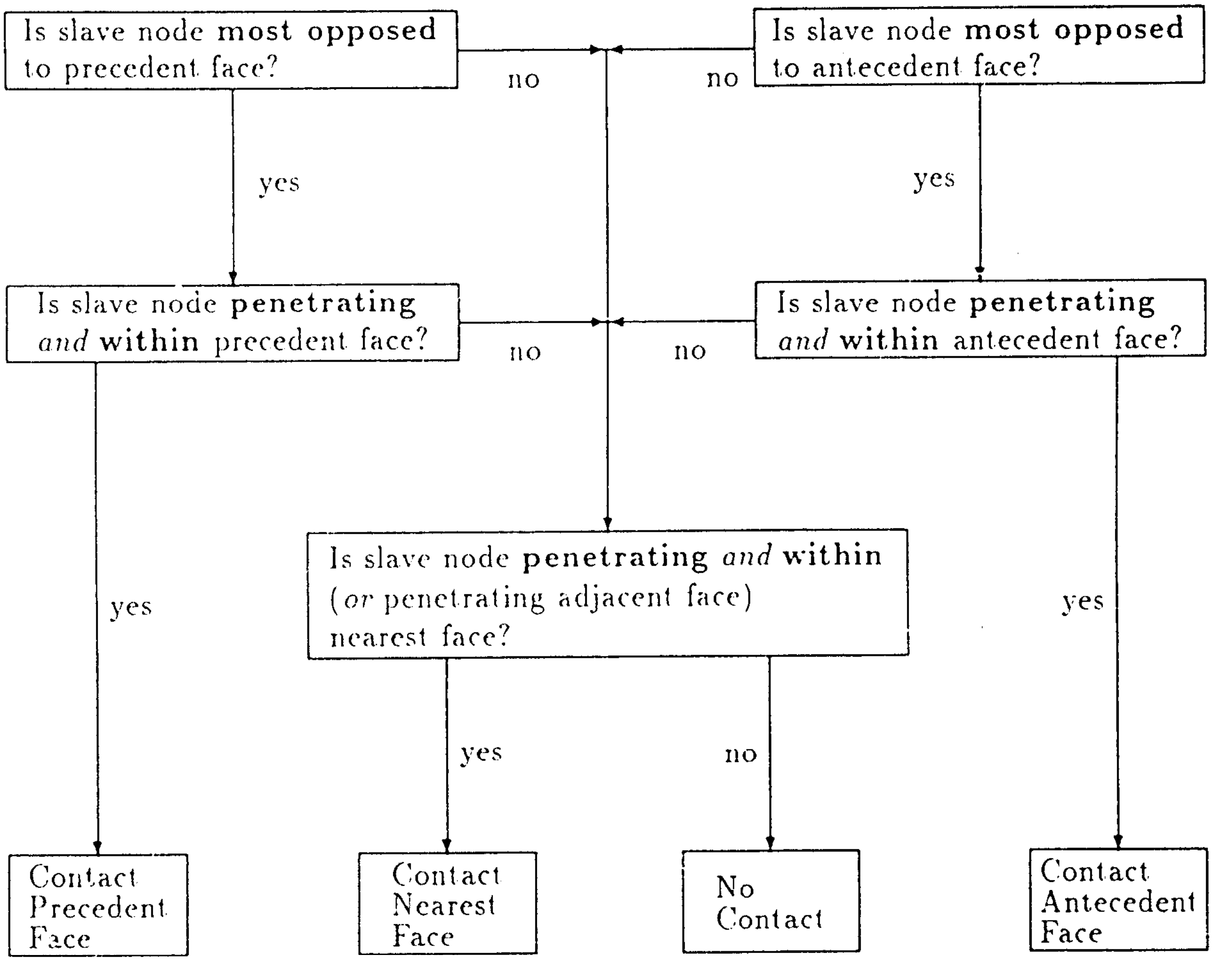

Is slave node **most opposed** to precedent face?
no
no
Is slave node **most opposed** to antecedent face?
yes
yes
Is slave node **penetrating** *and* **within** precedent face?
no
no
Is slave node **penetrating** *and* **within** antecedent face?
yes
Is slave node **penetrating** *and* **within** (*or* penetrating adjacent face) nearest face?
yes
yes
no
yes
Contact Precedent Face
Contact Nearest Face
No Contact
Contact Antecedent Face

Figure 6.4. Contact Determination Logic

6.2.5 Contact Forces

We use a partitioned kinematic approach to enforce compliance between two contact surfaces. This means that each surface acts as a master for a fraction of each time step and as a slave for the remainder.

The first task in restoring compliance is to calculate the penetration forces imposed on the master surface by the slave surface. We define these forces as a fraction of the forces which would be imposed by the slave nodes if the master surface was rigid. This fraction is the partition factor β, which represents the fraction of each time step for which these surfaces act as master and slave, respectively. Their roles are reversed for the remaining fraction $(1 - \beta)$.

The penetration force for a slave node is expressed by

$$\mathbf{f}_p = \frac{\beta m_S \delta}{\Delta t^2}\mathbf{n} \tag{6.62}$$

where m_S is the mass of the slave node. The penetration depth, δ, and vector, $\mathbf{n}$, are defined in Equations 6.59–6.60.

Next we want to find the response of the master surface to these penetration forces, such that the response of each contacting slave node is constrained by its master nodes as shown below.

$$\begin{aligned} \mathbf{a}_{pS} &= (\tfrac{1}{2} - \xi)(\tfrac{1}{2} - \eta)\mathbf{a}_{n1} \\ &+ (\tfrac{1}{2} + \xi)(\tfrac{1}{2} - \eta)\mathbf{a}_{n2} \\ &+ (\tfrac{1}{2} + \xi)(\tfrac{1}{2} + \eta)\mathbf{a}_{n3} \\ &+ (\tfrac{1}{2} - \xi)(\tfrac{1}{2} + \eta)\mathbf{a}_{n4} \end{aligned} \tag{6.63}$$

where $\mathbf{a}_{pS}$, and $\mathbf{a}_{n1}$ thru $\mathbf{a}_{n4}$ are the acceleration responses of the slave node and master nodes, respectively.

Equation 6.63 couples the response of individual master nodes. The principle of virtual work is applied to generate the following equations which define the accelerations of the master nodes in response to the penetration forces.

$$(m_I + \sum_S m_{I_S})\mathbf{a}_{nI} = \sum_S \mathbf{f}_{I_S} \tag{6.64}$$

where the summation is over all slave nodes S in contact with master node I.

The above expression represents a set of uncoupled equations; one for each master node. The mass and force contributions to the above assembly for a given slave node are as follows:

$$
\begin{aligned}
m_{1_S} &= (\tfrac{1}{2} - \xi)(\tfrac{1}{2} - \eta)m_S & (6.65)\\
m_{2_S} &= (\tfrac{1}{2} + \xi)(\tfrac{1}{2} - \eta)m_S & (6.66)\\
m_{3_S} &= (\tfrac{1}{2} + \xi)(\tfrac{1}{2} + \eta)m_S & (6.67)\\
m_{4_S} &= (\tfrac{1}{2} - \xi)(\tfrac{1}{2} + \eta)m_S & (6.68)
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{f}_{1_S} &= (\tfrac{1}{2} - \xi)(\tfrac{1}{2} - \eta)\mathbf{f}_p & (6.69)\\
\mathbf{f}_{2_S} &= (\tfrac{1}{2} + \xi)(\tfrac{1}{2} - \eta)\mathbf{f}_p & (6.70)\\
\mathbf{f}_{3_S} &= (\tfrac{1}{2} + \xi)(\tfrac{1}{2} + \eta)\mathbf{f}_p & (6.71)\\
\mathbf{f}_{4_S} &= (\tfrac{1}{2} - \xi)(\tfrac{1}{2} + \eta)\mathbf{f}_p & (6.72)
\end{aligned}
$$

After assembling and solving Equations 6.64 for the master accelerations, each slave response is interpolated via Equation 6.63. Note that this slave response restricts the motion induced by the penetration force given in Equation 6.62. Therefore, the acceleration correction for a slave node is given by

$$\mathbf{a}_{nS} = \mathbf{a}_{pS} - \frac{\mathbf{f}_p}{m_S} \; . \tag{6.73}$$

In the absence of friction, the corrected nodal acceleration to Equation 6.16 for both master and slave nodes then is given by

$$\mathbf{a} = \hat{\mathbf{a}} + \mathbf{a}_n \; . \tag{6.74}$$

6.2.6 Friction

Friction resists the relative tangential motion of the contacting slave nodes. The relative predicted tangential velocity of the slave node with respect to the master surface is calculated as follows:

$$
\begin{aligned}
\mathbf{v}_r = \hat{\mathbf{v}}_S &- (\tfrac{1}{2} - \xi)(\tfrac{1}{2} - \eta)\hat{\mathbf{v}}_1\\
&- (\tfrac{1}{2} + \xi)(\tfrac{1}{2} - \eta)\hat{\mathbf{v}}_2\\
&- (\tfrac{1}{2} + \xi)(\tfrac{1}{2} + \eta)\hat{\mathbf{v}}_3\\
&- (\tfrac{1}{2} - \xi)(\tfrac{1}{2} + \eta)\hat{\mathbf{v}}_4 \qquad (6.75)
\end{aligned}
$$

$$\mathbf{v}_s = \mathbf{v}_r - (\mathbf{n} \cdot \mathbf{v}_r)\mathbf{n} \tag{6.76}$$

The above velocity is decomposed into a magnitude and unit direction vector as follows:

$$v_s = \sqrt{\mathbf{v}_s \cdot \mathbf{v}_s} \tag{6.77}$$

$$\mathbf{s} = \frac{\mathbf{v}_s}{v_s} \tag{6.78}$$

As with the penetration force 6.62, we define the tangential contact force as a fraction of the force which must be applied to the slave node to cancel its relative tangential velocity. This force is given by

$$f_s = -\frac{\beta m_S v_s}{\Delta t} \tag{6.79}$$

where the minus sign above reflects that this force would be applied in the direction of s, but opposing the motion.

PRONTO currently supports three options for friction: no friction, Coulomb friction with a constant coefficient of friction, or the velocity-dependent friction law found in HONDO II [32]. The coefficient of friction can be expressed by

$$\mu = \mu_\infty + (\mu_0 - \mu_\infty)e^{-\gamma v_s} \tag{6.80}$$

where μ_0 and μ_∞ are the low- and high-velocity friction coefficients, respectively, and γ is a decay constant. Clearly, if γ equals zero, the coefficient of friction is the constant μ_0. Futhermore, if μ_0 is also equal to zero, the surface will be frictionless.

The magnitude of the tangential force exerted by the master surface on a slave node cannot exceed the maximum friction force. This constraint is expressed as

$$f_f = \frac{f_s}{|f_s|} \min(\mu f_n, |f_s|) \tag{6.81}$$

where f_n is the magnitude of the normal contact force as given below.

$$f_n = m_S \mathbf{a}_{nS} \cdot \mathbf{n} \tag{6.82}$$

Substituting Equations 6.82 and 6.79 into 6.81, then eliminating the nodal mass yields

$$a_s = \frac{v_s}{|v_s|} \min(\mu \mathbf{a}_{nS} \cdot \mathbf{n}, \frac{\beta |v_s|}{\Delta t}) \tag{6.83}$$

Applying this force to the slave node and balancing forces to the master nodes, then dividing by the appropriate nodal mass yields the following expressions for the tangential accelerations to these respective nodes.

$$\mathbf{a}_{sS} = a_s \mathbf{s} \tag{6.84}$$

$$\mathbf{a}_{s1} = -(\tfrac{1}{2} - \xi)(\tfrac{1}{2} - \eta)\frac{a_s m_S}{m_1}\mathbf{s} \tag{6.85}$$

$$\mathbf{a}_{s2} = -(\tfrac{1}{2} + \xi)(\tfrac{1}{2} - \eta)\frac{a_s m_S}{m_2}\mathbf{s} \tag{6.86}$$

$$\mathbf{a}_{s3} = -(\tfrac{1}{2} + \xi)(\tfrac{1}{2} + \eta)\frac{a_s m_S}{m_3}\mathbf{s} \tag{6.87}$$

$$\mathbf{a}_{s4} = -(\tfrac{1}{2} - \xi)(\tfrac{1}{2} + \eta)\frac{a_s m_S}{m_4}\mathbf{s} \tag{6.88}$$

Finally, by adding the above tangential accelerations to Equation 6.74, the corrected total acceleration of each contact node is expressed in general form for both master and slave nodes by

$$\mathbf{a} = \hat{\mathbf{a}} + \mathbf{a}_n + \mathbf{a}_s \tag{6.89}$$

7. BOUNDARY CONDITIONS

PRONTO supports several types of boundary conditions. In this chapter, we describe how these are implemented in the program. In Chapter 8, we discuss the initialization and time-stepping algorithm including when these various boundary conditions are applied.

7.1 Kinematic Boundary Conditions

The kinematic boundary conditions described below are *all* accomplished by altering the accelerations of the nodal points. All of the kinematic boundary conditions apply to nodal point sets.

7.1.1 No Displacement Constraint

A no displacement constraint is accomplished by setting the acceleration of each selected node to zero.

Note: Velocity or acceleration constraints prescribed on a node will override a no displacement constraint.

7.1.2 Prescribed Velocity Constraint

A prescribed velocity constraint is accomplished by altering the nodal point acceleration such that when the accelerations are integrated once, they provide the proper value of the nodal velocity. The nodal value of acceleration for the time step is calculated by the program as

$$a_t = \frac{(v_{t+\Delta t} - v_t)}{\Delta t} \quad . \tag{7.1}$$

The velocity at the end of the time step is computed by

$$v_{t+\Delta t} = s\, f(t + \Delta t) \tag{7.2}$$

where s is the scale factor and $f(t)$ is the history function defined by the user. In Equation 7.1, the value of velocity at the beginning of the time increment, v_t, is the value computed by Equation 7.2 at the previous time increment.

Note: A prescribed acceleration constraint on a node will override a prescribed velocity constraint.

7.1.3 Prescribed Acceleration Constraint

A prescribed acceleration constraint is applied by the program by setting the nodal acceleration during the time increment to the value given by

$$a_t = s\,f(t) \tag{7.3}$$

where s is the scale factor and $f(t)$ is the history function defined by the user.

Note: A prescribed acceleration constraint will override any other kinematic constraint on the same node.

7.2 Traction Boundary Conditions

The boundary conditions described below apply external forces to selected nodes. The pressure and nonreflecting boundary conditions deal with element side sets, while the nodal force boundary condition applies to nodal point sets.

7.2.1 Pressure

The set of consistent nodal point forces arising from pressures distributed over an element side are defined via the principle of virtual work by

$$\delta u_{iI} f_{iI} = \delta u_{iI} \int_S \phi_I(-pn_i)dA \quad . \tag{7.4}$$

where the range of the lowercase subscripts is 3, while the range of uppercase subscripts is 4.

Since the virtual displacements are arbitrary, they may be eliminated to yield:

$$f_{iI} = -\int_S \phi_I p n_i dA \tag{7.5}$$

The most general pressure distribution we allow is mapped from nodal point pressure values via the isoparametric shape functions. The resulting expression for the consistent nodal forces is

$$f_{iI} = -p_J \int_S \phi_I \phi_J n_i dA \quad . \tag{7.6}$$

For the eight-node uniform stress element used in PRONTO, ϕ_I is given by

$$\phi_I = \tfrac{1}{4}\Sigma_I + \tfrac{1}{2}\xi\Lambda_{1I} + \tfrac{1}{2}\eta\Lambda_{2I} + \xi\eta\Gamma_I \tag{7.7}$$

The above integral involves 64 terms. Only terms with even powers of both ξ and η are nonzero. Of the the remaining 16 terms, 4 vanish due to the properties of the alternator. The final 12 terms are given below.

$$\begin{aligned} f_{iI} = -p_J e_{ijk} x_{jM} x_{kN} \int_{-\frac{1}{2}}^{\frac{1}{2}} \int_{-\frac{1}{2}}^{\frac{1}{2}} & \left(\tfrac{1}{2}\Sigma_I + \tfrac{1}{2}\xi\Lambda_{1I} + \tfrac{1}{2}\eta\Lambda_{2I} + \xi\eta\Gamma_I\right) \\ & \left(\tfrac{1}{2}\Sigma_J + \tfrac{1}{2}\xi\Lambda_{1J} + \tfrac{1}{2}\eta\Lambda_{2J} + \xi\eta\Gamma_J\right) \\ & \left(\tfrac{1}{2}\Lambda_{1M} + \eta\Gamma_M\right) \left(\tfrac{1}{2}\Lambda_{2N} + \eta\Gamma_N\right) d\xi d\eta \end{aligned} \qquad (7.12)$$

Integrating yields

$$\begin{aligned} f_{iI} = -\tfrac{1}{64} p_J e_{ijk} x_{jM} x_{kN} \Big[& \left(\Sigma_I\Sigma_J + \tfrac{1}{3}\Lambda_{1I}\Lambda_{1J} + \tfrac{1}{3}\Lambda_{2I}\Lambda_{2J} + \tfrac{1}{9}\Gamma_I\Gamma_J\right)\Lambda_{1M}\Lambda_{2N} \\ & + \left(\tfrac{1}{3}\Sigma_I\Lambda_{1J} + \tfrac{1}{3}\Lambda_{1I}\Gamma_J + \tfrac{1}{9}\Gamma_I\Lambda_{2J} + \tfrac{1}{9}\Lambda_{2I}\Gamma_J\right)\Lambda_{1M}\Gamma_N \\ & + \left(\tfrac{1}{3}\Sigma_I\Lambda_{2J} + \tfrac{1}{3}\Lambda_{2I}\Gamma_J + \tfrac{1}{9}\Gamma_I\Lambda_{1J} + \tfrac{1}{9}\Lambda_{1I}\Gamma_J\right)\Gamma_M\Lambda_{2N} \Big] \end{aligned} \qquad (7.13)$$

The above expression may be evaluated to yield the following formula for calculating the nodal forces:

$$\begin{aligned} \begin{bmatrix} F_{i1} \\ F_{i2} \\ F_{i3} \\ F_{i4} \end{bmatrix} = & \tfrac{1}{72}\left[(x_{j3} - x_{j1})(x_{k4} - x_{k2}) + (x_{j2} - x_{j4})(x_{k3} - x_{k1})\right] \begin{bmatrix} 2(2p_1 + p_2 + p_4) + p_3 \\ 2(2p_2 + p_3 + p_1) + p_4 \\ 2(2p_3 + p_4 + p_2) + p_1 \\ 2(2p_4 + p_1 + p_3) + p_2 \end{bmatrix} \\ & + \tfrac{1}{72}\left[(x_{j2} - x_{j1})(x_{k3} - x_{k4}) + (x_{j3} - x_{j4})(x_{k1} - x_{k2})\right] \begin{bmatrix} -(2p_1 + p_4) \\ (2p_2 + p_3) \\ (2p_3 + p_2) \\ -(2p_4 + p_1) \end{bmatrix} \\ & + \tfrac{1}{72}\left[(x_{j1} - x_{j4})(x_{k3} - x_{k2}) + (x_{j3} - x_{j2})(x_{k4} - x_{k1})\right] \begin{bmatrix} -(2p_1 + p_2) \\ -(2p_2 + p_1) \\ (2p_3 + p_4) \\ (2p_4 + p_3) \end{bmatrix} \end{aligned} \qquad (7.14)$$

where $\{i, j, k\}$ form a negative permutation. Note that a positive pressure gives forces directed inward.

The nodal values for the pressure are calculated using the user-supplied scale factor and time history function. The values are calculated at the beginning of the time step.

The application of the pressure boundary conditions is fully vectorized. Element sides are processed in vector blocks using the vector block scratch element space. After the consistent nodal point forces are calculated for a block of element sides, they are accumulated into the global nodal force array.

7.2.2 Moving Pressures

The moving pressure boundary condition implemented in PRONTO represents a relatively simple way of incorporating both a spatial and temporal distribution of pressure loading on a surface. The implementation described here is intended for blast type loading on a surface where the blast originates from some point defined by the coordinates (x_0, y_0, z_0) and propagates along the surface. We assume that the surface is flat and the distance from any point on the surface to point (x_0, y_0, z_0) is given by d. Then the pressure at any point is written as

$$p(\tau, d) = a\tau e^{-b\tau} \tag{7.15}$$

where τ is the time measured from the arrival of the pressure wave at the point and a and b are functions of distance, which are defined below. If w is the propagation speed of the pressure wave along the surface, then τ is given by

$$\tau = t_0 + \frac{d}{w} \tag{7.16}$$

where t_0 is the pressure initiation time at the point (x_0, y_0, z_0). The time at which Equation 7.15 gives a maximum for the pressure is

$$\tau_{max} = \frac{1}{b} \tag{7.17}$$

which we refer to as the rise time. The peak pressure obtained at this time is

$$p_{max} = \frac{a}{b}e^{-1} \quad . \tag{7.18}$$

We allow the user to define two functions of distance from the point (x_0, y_0, z_0) which describe the behavior of the pressure wave. The first function defines the peak pressure as a function of distance while the second describes the rise time as a function of distance. Using Equations 7.17 and 7.18, we can write the parameters a and b as functions of distance

$$a(d) = \frac{f_1(d)}{f_2(d)}e^{+1} \quad , \quad b(d) = \frac{1}{f_2(d)} \quad . \tag{7.19}$$

The user can define the functions in any manner he sees fit, which allows for a quite general specification of the moving pressure wave. If the user inputs a zero value for the propagation speed, w, the code assumes that the pressure is applied instantaneously along the surface (i.e., this corresponds to an infinite propagation speed). If the assumed pressure description given by Equation 7.15 is not suitable, it is a simple task to change this description to some other two parameter functional form and alter the code accordingly.

7.2.3 Nodal Forces

Nodal point external forces are applied by calculating the magnitude of the force determined by the user supplied scale factor and time history function. The time history function is evaluated at the beginning of the time step.

7.3 Nonreflecting Boundaries

In a number of geotechnical applications, it is desirable to model an infinite or semi-infinite space. In these applications, waves are transmitted outward from some disturbance and are absorbed in the far field. PRONTO contains a boundary condition specification which will absorb waves and suppress any reflection back into the interior mesh. This allows for a much smaller mesh and a significant reduction in the number of degrees of freedom in the problem.

The absorbing or nonreflecting boundary which is implemented in the code was proposed by Lysmer and Kuhlemeyer [33], and discussed in detail by Cohen and Jennings [34]. The exterior infinite region is replaced by an energy absorbing boundary condition. The basic idea is to apply boundary tractions which will exactly cancel the stresses which are generated at the free surface. On this boundary surface, tractions are applied of the form

$$\sigma_n = \rho V_p \dot{u}_n \tag{7.20}$$

and

$$\tau_s = \rho V_s \dot{u}_t \tag{7.21}$$

where:

σ_n = normal stress applied to the boundary
τ_s = shear stress applied to the boundary
$\dot{u}_n$ = velocity component normal to the boundary
$\dot{u}_t$ = velocity component tangential to the boundary
ρ = current density of the material at the boundary
V_s = current s-wave velocity in the material at the boundary
V_p = current p-wave velocity in the material at the boundary

The wave speeds required in Equations 7.20 and 7.21 are calculated using the effective shear and dilatational moduli determined in Section 3.4.

The tractions given by Equations 7.20 and 7.21 are used with the consistent nodal forces which were derived in Section 7.2.1. The normal and tangential nodal velocities are determined for the four local nodes of the element side to determine the correct tractions. These are used in Equation 7.14 for both the normal and shear components to give the proper consistent nodal point forces for the absorbing boundary.

The application of the nonreflecting boundary condition is vectorized in a manner similar to the pressure boundary condition. The effective moduli required for the wave speed determinations in Equations 7.20 and 7.21 are computed during the main element loop and stored for elements having this boundary condition.

8. INITIALIZATION AND TIME-STEPPING ALGORITHM

8.1 Initialization

The user defines a mechanics problem by specifying material properties, body geometry, initial conditions, tractions, and kinematic constraints. PRONTO does an extensive amount of data checking to try to insure that the user has defined a valid mechanics problem. These checks range from mundane (e.g., Are positive mass densities provided for the materials?) to more subtle (e.g., Are all the contact surfaces simply connected?). We do not guarantee that PRONTO will always detect bad data, but experience has shown that it is usually smarter than both the users and the authors.

By "initialization" we mean the calculations which must be performed and the data structures which must be set up before entering the time-stepping loop. There are two initialization processes in PRONTO. The first has to do with setting up the initial data structures according to user specifications. This is all done in the INIT routine which is called from the main program. The initial displacements and velocities are first set to zero. Then the initial velocities defined by the user are set. The stresses and internal state variables also are initialized to zero. The internal state variables are subsequently reset to the appropriate initial values for the respective material model, if necessary. Construction of surface data structures and the initial tracking of the contact surfaces are also performed within the INIT routine.

The second, and more subtle, initialization which must be performed concerns the resolution of the initial velocity field defined by the user with the kinematic constraints. A pseudo-time step is performed to force the initial velocity field to be kinematically compliant. It is important to enforce the initial constraints before entering the element processing loop in order to allow the shock viscosity (quadratic bulk viscosity) to respond.

To illustrate the necessity of this operation, consider the case of a bar striking a rigid wall at some nonzero initial velocity. The user defines a rigid surface and assigns all of the material in the bar the same initial velocity towards the wall. But the nodes initially in contact with the wall would then violate the rigid surface constraints. To restore kinematic compliance, the initial velocity of these nodes would have to be reset to zero. This induces a high initial strain rate in the first row of elements.

The pseudo-time step is performed within the SOLVE routine prior to entering the time step loop. It uses the initial time increment determined in the INIT routine during the element mass calculations. The algorithm proceeds as follows:

1. Set the pseudo-time accelerations equal to the trial velocities divided by the initial time increment. Then set the velocities to zero.

2. Predict a new configuration based on these pseudo-time accelerations.

3. Calculate acceleration corrections which enforce all kinematic constraints, *except prescribed accelerations.*

4. Reset the initial velocities equal to the corrected pseudo-time accelerations multiplied by the initial time increment, and reset the current coordinates to the original configuration. The initial velocity field is now kinematically compliant.

We translate the trial velocities to pseudo-time accelerations strictly for algorithmic convenience. *All* kinematic constraints are enforced by PRONTO via accelerations. Therefore, the pseudo-time accelerations can use the same code as the main time step loop to enforce constraints.

PRONTO next enters the time step loop. The initial time step is performed the same way as all subsequent steps, *except that the constitutive state is not updated!*

8.2 Time Step Loop

The order of operations within the time step loop is crucial to ensuring correct results. PRONTO's time-stepping algorithm proceeds in the following order:

1. Assemble the internal forces from the following:

    (a) Calculate element strain rates.

    (b) Advance the constitutive state to the end of the time step.

    (c) Calculate force contributions due to stress divergence, artificial viscosity, and hourglass resistance. Determine the new stable time increment.

2. Apply external loads: pressures, nonreflecting boundaries, and nodal forces.

3. Calculate accelerations and predict the configuration at the end of the time step ignoring kinematic constraints. This predicted configuration will be used in the contact and rigid surface routines to determine the corrections which must be made to the accelerations to bring the surfaces back into compliance.

4. Apply kinematic constraints by altering the accelerations to satisfy prescribed: displacement, velocity, and acceleration.
5. Enforce rigid surface and contact surface constraints by altering the accelerations, *without disturbing the consiraints enforced above.*
6. Write output, if timely.
7. Integrate the velocities and displacements forward using the compliant accelerations. Update the current spatial coordinates.
8. Update the current time and go back to step 1 if more time is required. Otherwise, exit the time step loop.

Most of the computation time in PRONTO occurs in step 1. Elements are processed in material and vector blocks during this phase. See Appendix B for further description of this process.

9. NUMERICAL EXAMPLES

In this chapter we present two representative example problems which demonstrate many of the features found in PRONTO 3D. In addition, all the example problems presented in the PRONTO 2D [1] manual were duplicated using the three-dimensional code.

9.1 Sphere Impact

This example problem is intended to demonstrate the robust nature of the contact surface algorithms described in Chapter 6, as well as the extremely large deformations and distortions which the uniform strain hexahedron can perform. Figure 9.1 shows a 1cm thick plate which is made of rolled homogeneous armor (RHA). The plate is impacted by a sphere of radius 2cm which is made of Staballoy. Table 9.1 lists the elastic/plastic combined hardening material properties used in the analysis. The impact velocity is 1000 m/sec at an angle of attack of 30 degrees.

Table 9.1. Sphere Impact Problem Material Properties

	RHA	Staballoy
Density	7800 kg/m^3	18,620 kg/m^3
Young's Modulus	206.8 Gpa	195.8 Gpa
Poisson's Ratio	0.3	0.203
Yield Stress	1220 Mpa	1036 Mpa
Hardening Modulus	1220 Mpa	1036 Mpa
Beta	0.5	0.5

There are 9052 elements and 10897 nodes in the mesh. The problem ran for 94.4 cpu seconds on a CRAY X-MP 4/16 under CTSS with CFTLIB. The total number of time steps was 411; this yields a value of 25.4 microseconds per element cycle.

Figure 9.2 shows a sequence of four different times during the impact event. The analysis was run to a time of 8.25 microseconds, at which time an element turned inside out and the code stopped. This represents the limit that the Lagrangian code can continue to model the event. Note the severe distortion in the elements which

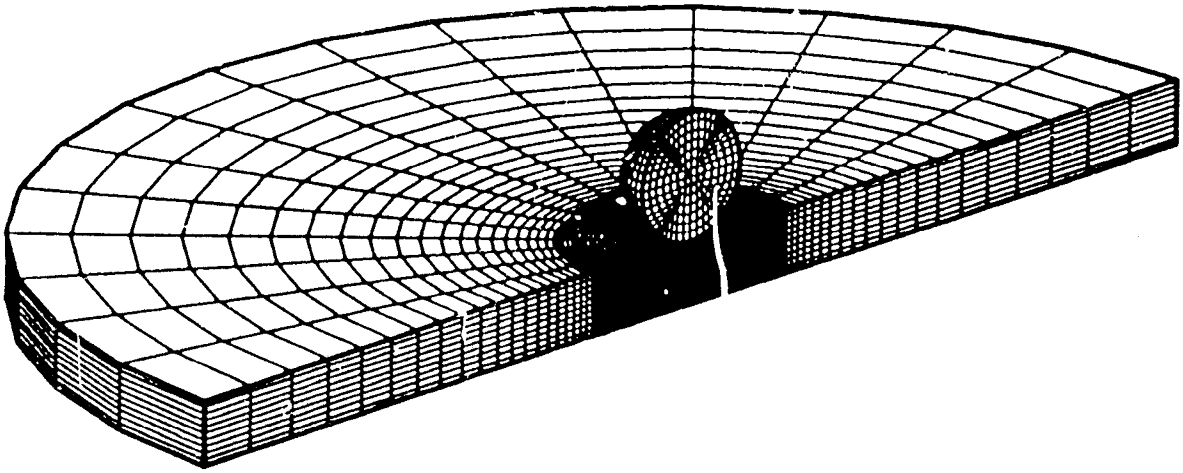


Figure 9.1. Sphere Impact Problem Mesh

form the impact crater. This distortion poses a tremendous challenge to the contact surface algorithm because element faces along the edge of the crater become extremely warped. Figure 9.3 shows views of the formation of the impact crater at the same four times as in Figure 9.2.

Figure 9.4 shows the PRONTO 3D commands used to define this problem.

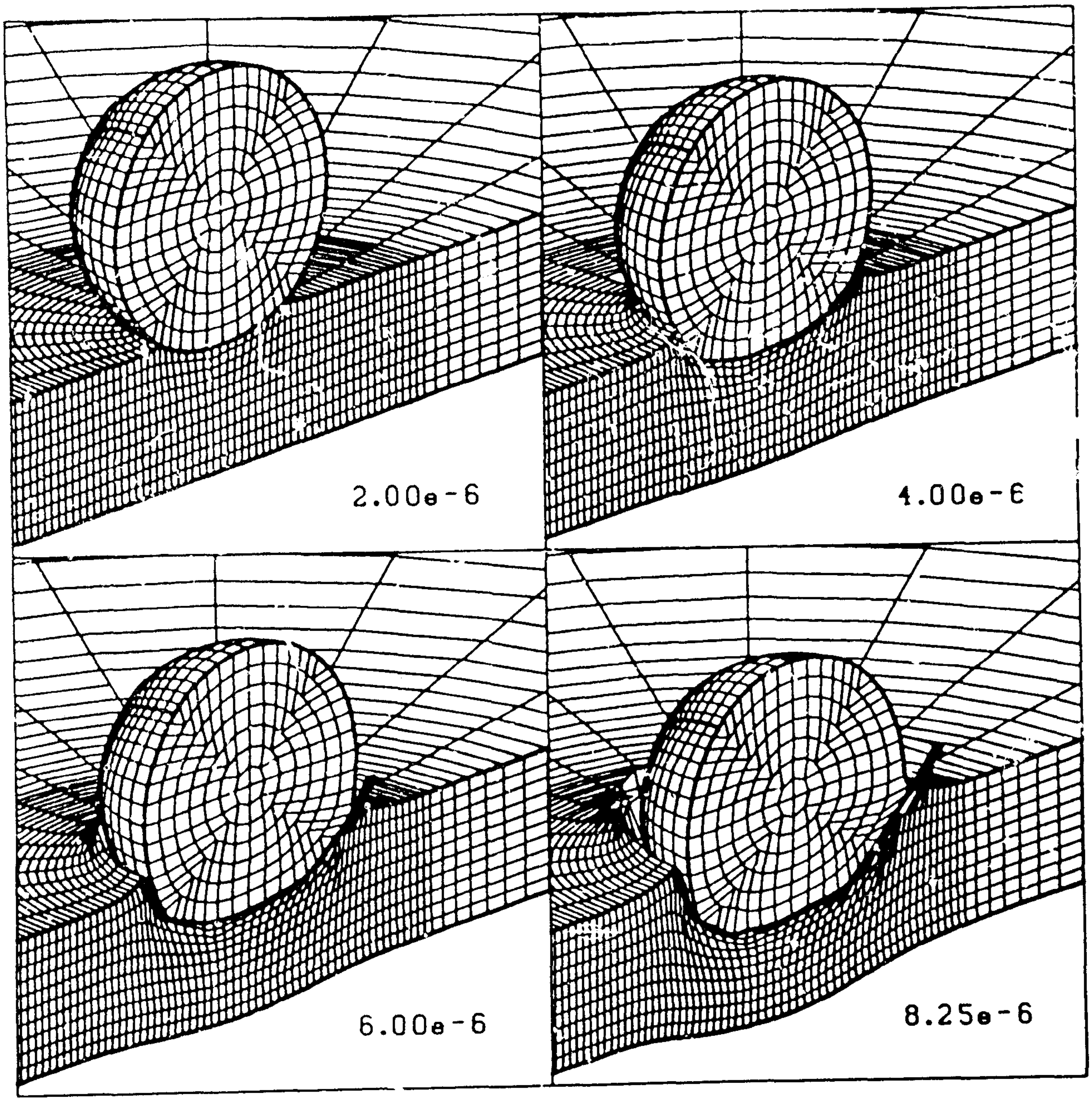


Figure 9.2. Time Sequence from the Sphere Impact Problem

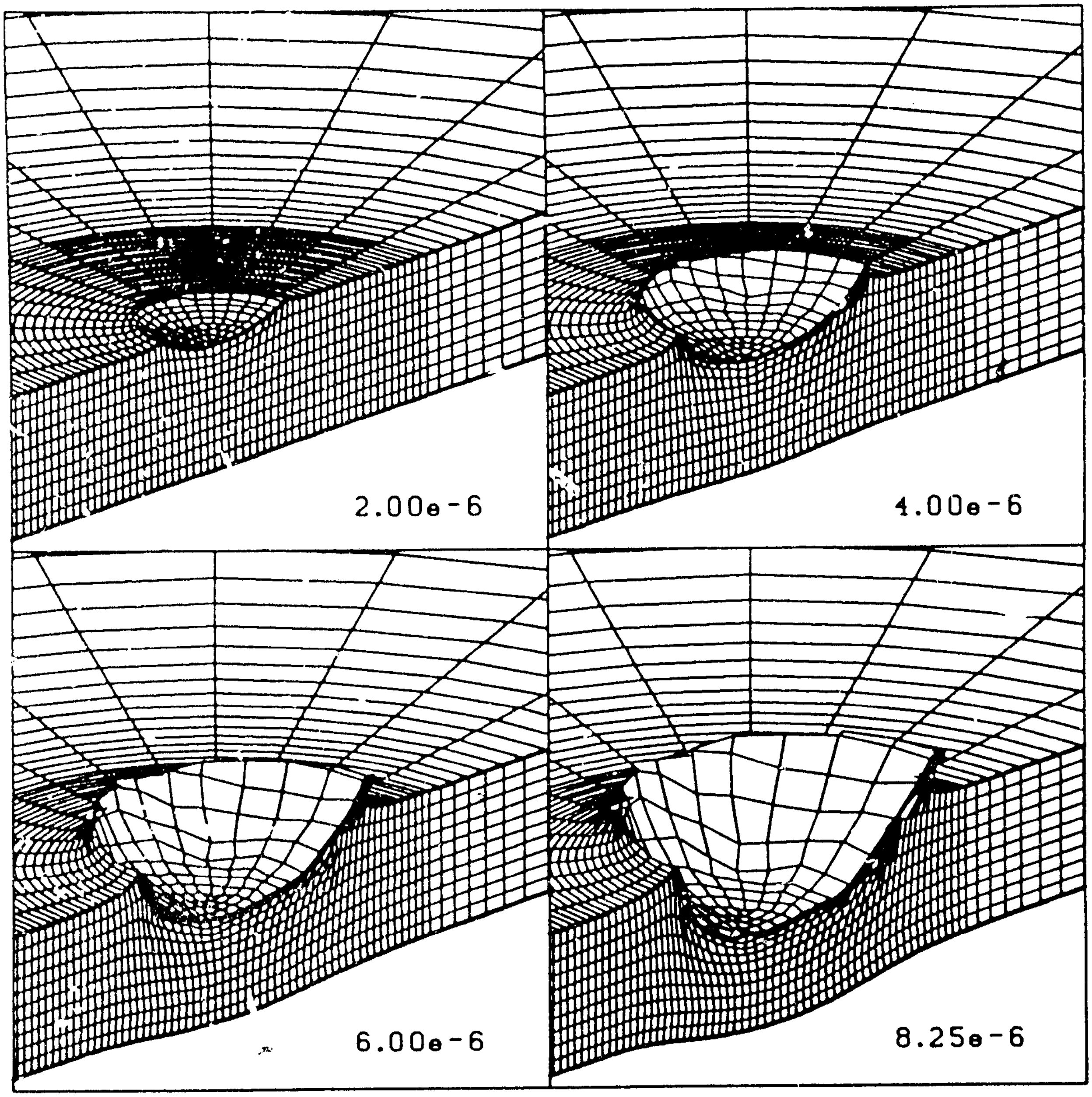
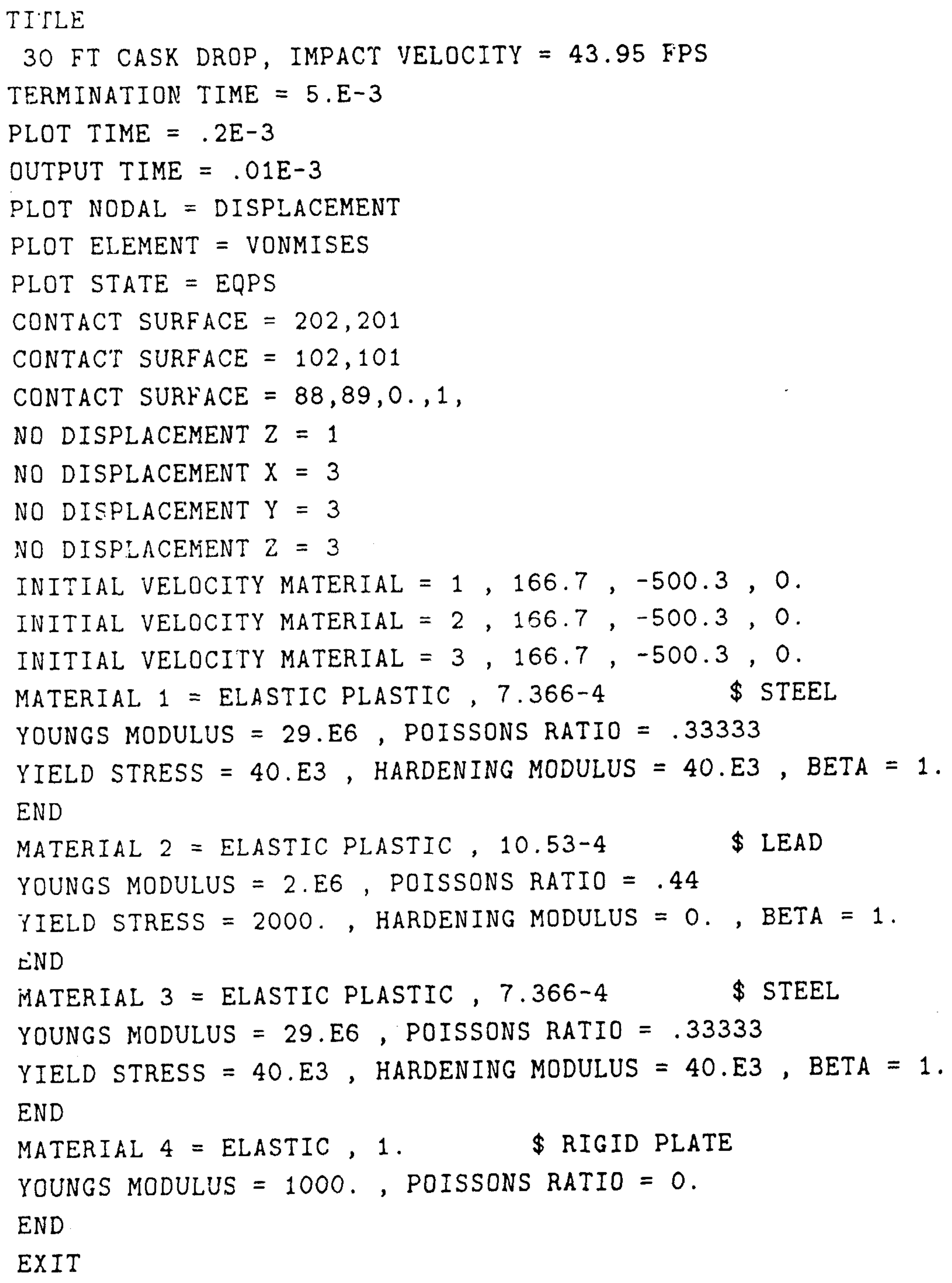


Figure 9.3. Impact Crater Formation during the Sphere Impact Problem



```
TITLE
 30 FT CASK DROP, IMPACT VELOCITY = 43.95 FPS
TERMINATION TIME = 5.E-3
PLOT TIME = .2E-3
OUTPUT TIME = .01E-3
PLOT NODAL = DISPLACEMENT
PLOT ELEMENT = VONMISES
PLOT STATE = EQPS
CONTACT SURFACE = 202,201
CONTACT SURFACE = 102,101
CONTACT SURFACE = 88,89,0.,1,
NO DISPLACEMENT Z = 1
NO DISPLACEMENT X = 3
NO DISPLACEMENT Y = 3
NO DISPLACEMENT Z = 3
INITIAL VELOCITY MATERIAL = 1 , 166.7 , -500.3 , 0.
INITIAL VELOCITY MATERIAL = 2 , 166.7 , -500.3 , 0.
INITIAL VELOCITY MATERIAL = 3 , 166.7 , -500.3 , 0.
MATERIAL 1 = ELASTIC PLASTIC , 7.366-4        $ STEEL
YOUNGS MODULUS = 29.E6 , POISSONS RATIO = .33333
YIELD STRESS = 40.E3 , HARDENING MODULUS = 40.E3 , BETA = 1.
END
MATERIAL 2 = ELASTIC PLASTIC , 10.53-4        $ LEAD
YOUNGS MODULUS = 2.E6 , POISSONS RATIO = .44
YIELD STRESS = 2000. , HARDENING MODULUS = 0. , BETA = 1.
END
MATERIAL 3 = ELASTIC PLASTIC , 7.366-4        $ STEEL
YOUNGS MODULUS = 29.E6 , POISSONS RATIO = .33333
YIELD STRESS = 40.E3 , HARDENING MODULUS = 40.E3 , BETA = 1.
END
MATERIAL 4 = ELASTIC , 1.        $ RIGID PLATE
YOUNGS MODULUS = 1000. , POISSONS RATIO = 0.
END
EXIT
```

Figure 9.4. PRONTO 3D Input Commands for the Sphere Impact Problem

9.2 Cask Impact

In this example problem a generic waste transportation cask is dropped from 30 feet onto a rigid rail. The impact velocity is 43.95 feet per second. The angle of impact is such that the center of gravity of the cask is over the corner where the impact occurs. The mesh of the cask used in the analysis is shown in Figure 9.5. The cask has 0.5 inch thick steel inner and outer liners with 3.5 inches of lead shielding between them. Table 9.2 lists the elastic/plastic material properties used in the analysis.

Table 9.2. Cask Impact Problem Material Properties

	Steel	Lead
Density	7.366×10^{-4} lb/in/sec^2	10.53×10^{-4} lb/in/sec^2
Young's Modulus	29×10^6 psi	2×10^6 psi
Poisson's Ratio	.33	.44
Yield Stress	40,000 psi	2,000 psi
Hardening Modulus	40,000 psi	0
Beta	1.	1.

The analysis was run to a total time of 5 milliseconds. Figure 9.6 shows the total kinetic energy in the system. Rebound occurs at 4.6 milliseconds, at which time the deformations in the cask are the largest. The configuration at that time is shown in Figure 9.7. The deformations in this analysis are not extremely large, as can be seen in the figure. Nevertheless, the materials in the cask, particularly the lead shielding, develop large plastic strains as shown in Figure 9.8.

This problem is included because it entails a large amount of contact data. A contact surface is defined between the liners and the shielding, and between the outer liner and the rail. The three contact surfaces make this a highly contact intensive analysis. The problem took 10932 time steps and used a total of 3732 cpu seconds on a CRAY X-MP 4/16 under CTSS with CFTLIB. This gives a value of 32.7 microseconds of cpu time per element cycle. Comparing this number with the value of 25.4 for the previous problem, shows that the intensive contact calculations for this problem carry only a 29 percent penalty.

Figure 9.9 shows the PRONTO 3D commands used to define this problem.

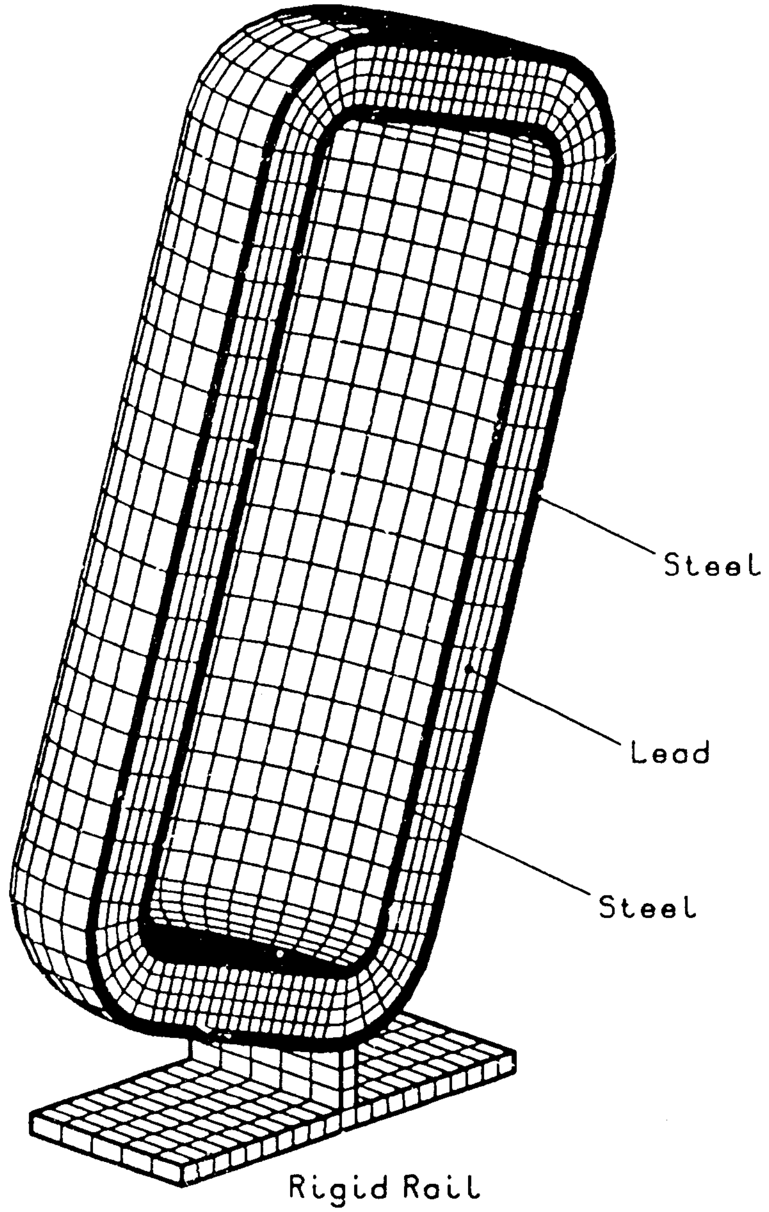


Figure 9.5. Cask Impact Problem Definition

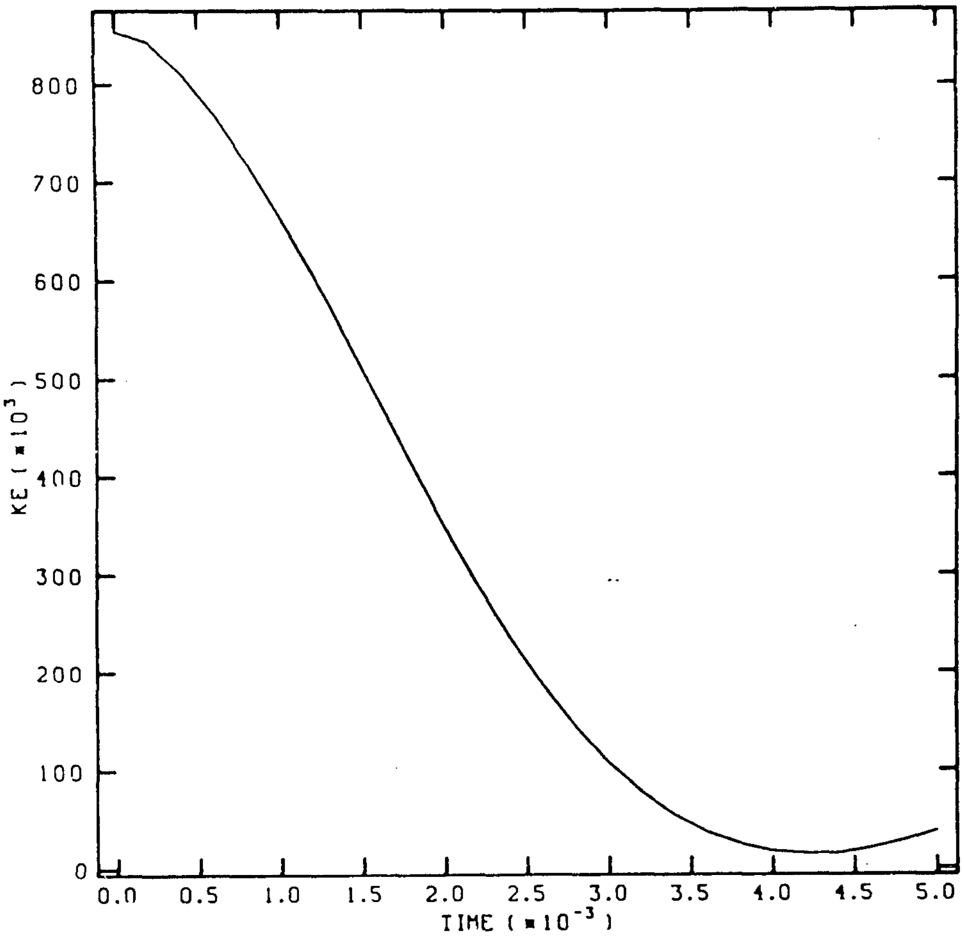


Figure 9.6. Kinetic Energy during the Cask Impact Problem

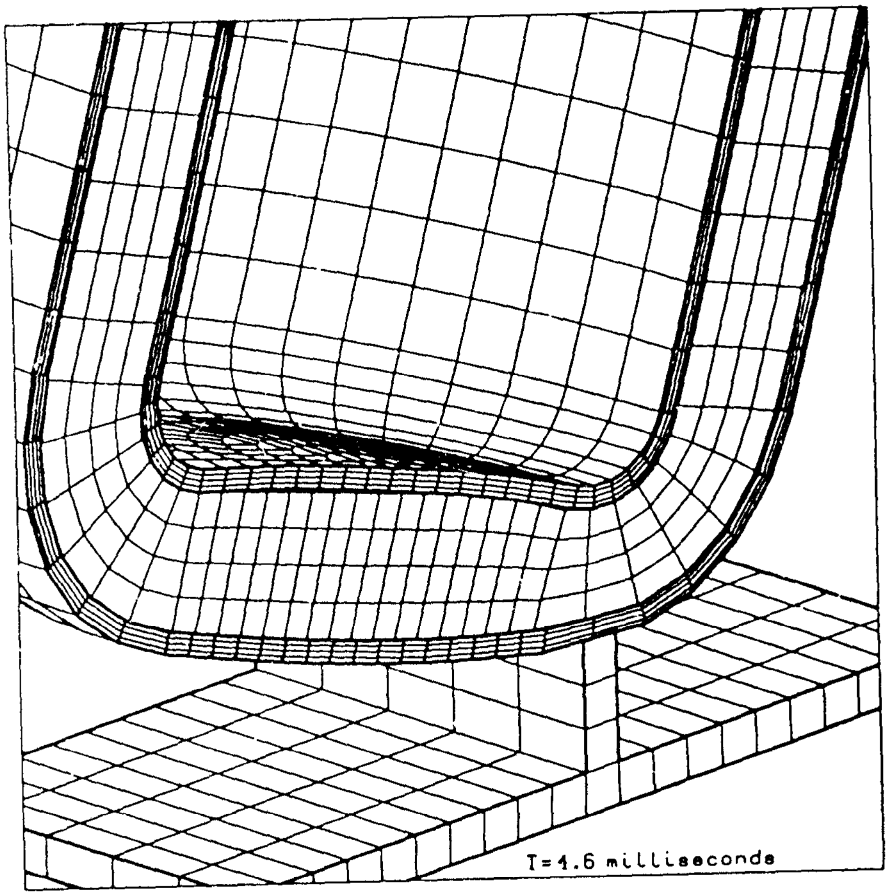


Figure 9.7. Deformed Shape at 4.6msec of Cask Impact Problem

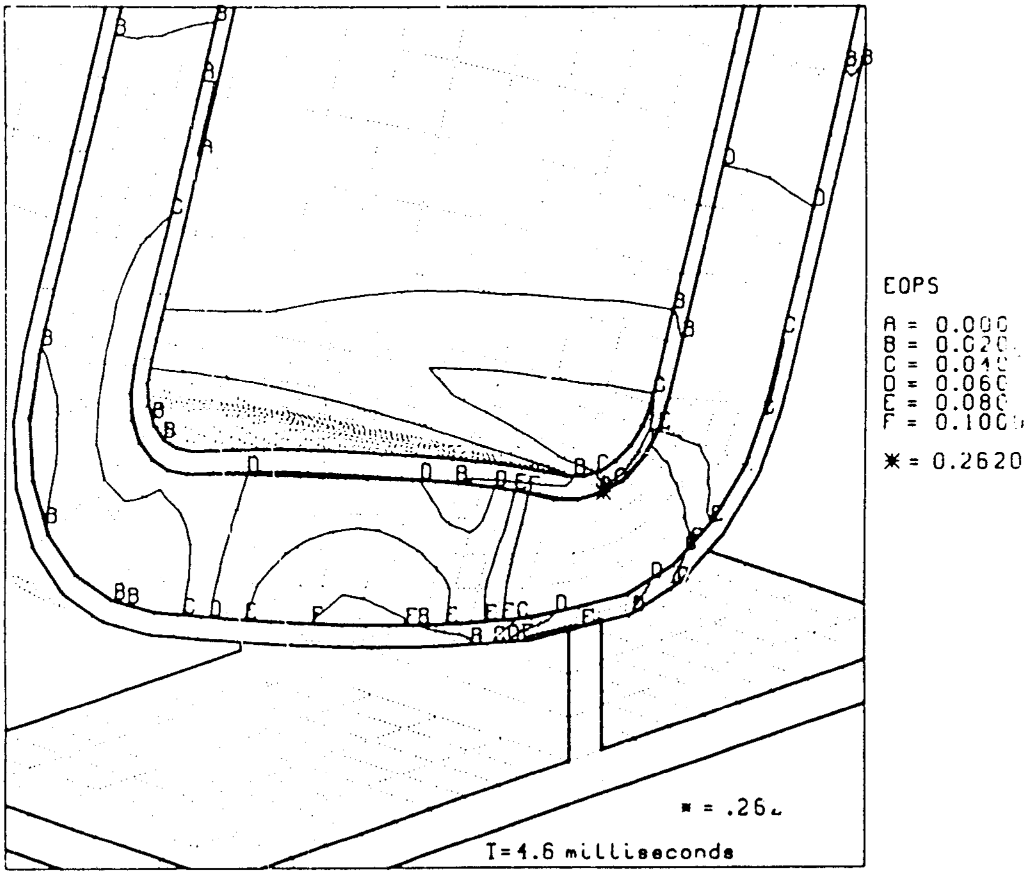


Figure 9.8. Equivalent Plastic Strain Developed in the Cask Impact Probiem

```
TITLE
  STEEL SPHERE IMPACTING STEEL PLATE AT 1000 M/SEC - ANGLE = 30 DEGREES
TERMINATION TIME = 8.25E-6
PLOT TIME = .25E-6
OUTPUT TIME = .25E-6
PLOT ELEMENT = VONMISES
PLOT STATE = EQPS
PLOT NODAL = DISPL
CONTACT SURFACE = 100 , 89
INITIAL VELOCITY MATERIAL = 1 , 500. , -866. , 0.
MATERIAL 2 = ELASTIC PLASTIC , 7800.  $ RHA
  YOUNGS MODULUS = 206.8E9
  POISSONS RATIO= .3
  YIELD STRESS = 1220.E6
  HARDENING MODULUS = 1220.E6
  BETA = .5
END
MATERIAL 1 = ELASTIC PLASTIC , 18620.  $ Staballoy
  YOUNGS MODULUS = 195.8E9
  POISSONS RATIO= .203
  YIELD STRESS = 1036.E6
  HARDENING MODULUS = 1036.E6
  BETA = .5
END
NO DISPLACEMENT Y = 300
NO DISPLACEMENT Z = 333
EXIT
```

Figure 9.9. PRONTO 3D Input Commands for the Cask Impact Problem

A. PRONTO 3D USERS INSTRUCTIONS

Listed below are all the keywords supported by the PRONTO 3D command language. The uppercase letters represent the minimum abbreviation of each word.

1. TITle
2. TERMination TIme
3. OUTput TIme
4. READ REStart
5. WRIte REStart
6. PLOT TIme
7. PLOT NODal
8. PLOT ELement
9. PLOT STate
10. PLOT HIstory
11. TIme STep SCale
12. BULK VIScosity
13. HOURglass STIFfening
14. EXIT
15. FUNCtion
16. NO DISplacement
17. PREScribed VELocity
18. PREScribed ACCeleration
19. PREScribed FORce
20. INITial VELocity NODeset
21. INITial VELocity MATerial
22. INITial VELocity ANGular
23. PRESsure
24. MOVing PREssure
25. SILent BC
26. RIGid SURface
27. CONtact SURface
28. MATerial
29. EQuation OF STate
30. DETonation POint
31. BURN CONstant
32. DELete MATerial
33. DEATh

The user presents input data to PRONTO in a keyword driven, free field format command language. The command lines may be in any order the user finds convenient. Each command is described below in the order in which they are listed above. The boldface words are the command keywords. The words following each command represent input parameters which the user should specify. Default values are defined where appropriate. Example command files are shown in Chapter 9.

The free field parser allows the user to delimit entries by one or more spaces, a comma, or an equals sign. Consecutive delimiters define a null (defaulted) field. We suggest using spaces between keywords, an equals sign to separate the command from the parameter list, and commas between values. Material data is input via material cues; an equal sign is helpful to separate each cue from its associated value. See the examples following command 28.

A dollar sign allows the user to place a comment on any line; anything following a dollar sign on an input line is ignored. An asterisk at the end of an input line indicates that the line is continued on the next line. This input style is described in greater detail in [5].

1. **TITle**
   enter a suitable title on the *next* line

2. **TERMination TIme**
   tend   time to terminate the analysis

3. **OUTput TIme**, tout
   tout   time interval at which to print output
   (default = tend/200, where tend is defined via command 2)

4. **REAd REStart**, restm
   restm   time at which restart is to begin

5. **WRIte REStart**, trsdmp
   trsdmp   time interval at which to write restart dump files
   (default is to write no restart files)

6. **PLOT TIme**, tplot, tstart, tpend
   tplot   time interval at which to write EXODUS output
   (default = (tpend-tstart)/10)
   tstart   time to start writing data EXODUS output
   (default = 0)
   tpend   time to stop writing data EXODUS output
   (default = tend from command 2)

7. **PLOT NODal**, nodal variable 1, nodal variable 2, ...

allowable nodal variable names (can be abbreviated to 3 letters):

DISPLACEMENT	- displacements (DISPLX, DISPLY, DISPLZ)
VELOCITY	- velocities (VELX, VELY, VELZ)
ACCELERATION	- accelerations (ACCLX, ACCLY, ACCLZ)
REACTION	- reactions (RX, RY, RZ)
MASS	- lumped mass (MASS)

The nodal variables written to the EXODUS file by default are the displacements, the velocities, and the accelerations. The displacements are *always* written to the EXODUS file. The names in parentheses are the alphanumeric names assigned to the variables on the EXODUS file.

8. **PLOT ELement**, element variable 1, element variable 2, ...

allowable element variable names:

STRESS	- stresses (SIGXX, SIGYY, SIGZZ, TAUXY, TAUYZ, TAUZX)
ENERGY	- internal energy density (ENERGY)
STRAIN	- total strains (EPSXX, EPSYY, EPSZZ, EPSXY, EPSYZ, EPSZX)
RATEDFM	- deformation rates (DXX, DYY, DZZ, DXY, DYZ, DZX)
STRETCH	- left stretches (STRECHXX, STRECHYY, STRECHZZ, STRECHXY, STRECHYZ, STRECHZX)
ROTATION	- rotations (R11, R21, R31, R21, R22, R23, R31, R32, R33)
DENSITY	- current density (DENSITY)
PRESSURE	- pressure (PRESSURE)
VONMISES	- vonMises equivalent stress (VONMISES)
BULKQ	- bulk viscosity pressure (BULKQ)

The element variables written to the EXODUS file by default are the stresses and the energy density. The names in parentheses are the alphanumeric names assigned to the variables on the EXODUS file.

9. **PLOT STate**, state variable 1, state variable 2, ...

The user can ask for any of the internal state variables to be written on the EXODUS file. Table A.1 lists the internal state variable names for each material model. See Chapter 4 for definitions of these variables. There are no state variables written to the EXODUS file by default.

Table A.1. Internal State Variables Available for Each Material Model

MATERIAL	ALLOWABLE NAMES			
ELASTIC	(no internal state variables)			
ELASTIC PLASTIC	EQPS	RADIUS	ALPHA11	ALPHA22
	ALPHA33	ALPHA12	ALPHA23	ALPHA31
VISCOPLASTIC	EQPS	SIGYLD		
DAMAGE	EQPS	DAMAGE	EVMAX	FRAGSIZE
	CRKDENS			
HYDRO	(no internal state variables)			
LOW DEN FOAM	PAIR			
SOIL N FOAMS	EVMAX	EVFRAC	EV	NUM
EP HYDRODYNAMIC	EQPS	RADIUS	ALPHA11	ALPHA22
	ALPHA33	ALPHA12	ALPHA23	ALPHA31

WARNING: *Indiscriminate use of commands 6, 7, 8, and 9 can create extremely large EXODUS files!*

10. **PLOT HIstory,** VARIABLE = variable name, COORD = x_0, y_0, z_0,
NAME = user name, COMP - comp name

VARIABLE keyword which defines the variable to be placed on the data base. Can be a nodal, an element or a state variable name.

variable name any valid nodal, element, or state variable name
For nodal variables, the valid names (which may be abbreviated to three characters) are:

DISPLACMENT
VELOCITY
ACCELERATION
REACTION
MASS

For element variables the valid names (which may be abbreviated to four characters) are:

STRESS
STRAIN
STRECH
RATEDFM
ENERGY
DENSITY

PRESSURE
VONMISES
BULKQ

For state variable names, the valid names are listed in Table A.1.

COORD — keyword to define the position of the history point

x_0, y_0, z_0 — coordinates of the history point (3 data values). PRONTO will find the nearest node or element to this position.

NAME — keyword for naming the output variable

user name — user supplied history output name. Must be between one and six characters. If the component specification is omitted for a vector or tensor variable, PRONTO will construct names for all the components by using the supplied name and appending the vector or tensor component designations. The user name must be unique for each history point defined. Futhermore, the user name must not conflict with the nodal, element, or global names that PRONTO defines (e.g., NAME=SIG will conflict with the element stresses, while NAME=SIGA will be unique).

COMP — optional keyword that specifies the component of the vector or tensor. This option is not allowed for scalar variables (e.g. PRESSURE); this includes *all* state variables. If you do not specify a component for a vector or tensor variable, PRONTO will create history variables for all components.

comp name — Vector or tensor component specification
For vectors: X, Y, or Z
For tensors: XX, YY, ZZ, XY, YZ, or ZX

11. **TIme STep SCale**, scft, ssft

scft — scale factor to be applied to the internally calculated global time increment
(default=1.0)

scft — scale factor to be applied to the internally calculated time step for strain softening elements
(default=1.0)

12. **BULK VIScosity**, b1, b2

b1 — linear bulk viscosity coefficient
(default = .06)

b2 — quadratic bulk viscosity coefficient
(default = 1.2)

13. **HOURglass STIFfening**, hgstiff

hgstiff   hourglass stiffening factor
(default=.05)

14. **EXIT**

terminates command input; remaining lines in command file are ignored

15. **FUNCtion**, function id

function id   any nonzero integer by which you wish to identify this function. Each function must have a unique id.

After a FUNCTION command you must enter a list of points defining the function. Each abscissa-ordinate pair is input on a separate line immediately following the command line. The list is terminated by a line containing the END command, as shown below.

$x_1, f(x_1)$
$x_2, f(x_2)$
$\vdots$
$x_n, f(x_n)$
END

The abscissae of a function *must* increase monotonically. PRONTO linearly interpolates between function points, but does *not* extrapolate. If the argument to the function falls outside of the user specified range, PRONTO ignores the boundary condition or load associated with that function. This means that a boundary condition can turn on or off at a specific time .

16. **NO DISplacement**, direction, node set id

direction   X, Y, or Z
node set id   must match a node set on the GENESIS file

PRONTO enforces zero displacement in the specified direction for each node in this set.

17. **PREScribed VELocity**, direction, node set id, function id, scale factor, $c_x, c_y, c_z, n_x, n_y, n_z$

direction   X, Y, Z, RADIAL, CYLINDRICAL, NORMAL, or SPHERICAL
node set id   must match a node set on the GENESIS file
function id   must match a function defined via command 15
scale factor   scales the function value
(default=1.0)

c_x, c_y, c_z	center point coordinates, defined *only* for options: RADIAL, CYLINDRICAL, or SPHERICAL
n_x, n_y, n_z	axis or normal vector, defined *only* for opticns: RADIAL, CYLINDRICAL, or NORMAL

PRONTO sets the appropriate component of velocity of each node in this set to the product of the time function value and the scale factor. The RADIAL option defines the radial velocity component with respect to a cylindrical coordinates system defined by the center point and axis vector. The CYLINDRICAL option defines the tangential (counterclockwise) velocity with respect to this cylindrical coordinate system. The NORMAL option simply defines a Cartesian component in a direction which is not aligned with one of the coordinate axes. Finally, the SPHERICAL option defines the radial velocity with respect to a spherical coordinate system.

18. **PREScribed ACCeleration**, direction, node set id, function id, scale factor

direction	X, Y, or Z
node set id	must match a node set on the GENESIS file
function id	must match a function defined via command 15
scale factor	scales the function value (default = 1.0)

PRONTO sets the specified component of acceleration of each node in this set to the product of the time function value and the scale factor.

19. **PREScribed FORce**, direction, node set id, function id, scale factor, $c_x, c_y, c_z, n_x, n_y, n_z$

direction	X, Y, Z, RADIAL, CYLINDRICAL, NORMAL, or SPHERICAL
node set id	must match a node set on the GENESIS file
function id	must match a function defined via command 15
scale factor	scales the function value (default=1.0)
c_x, c_y, c_z	center point coordinates, defined *only* for options: RADIAL, CYLINDRICAL, or SPHERICAL
n_x, n_y, n_z	axis or normal vector, defined *only* for options: RADIAL, CYLINDRICAL, or NORMAL

PRONTO applies the appropriate force component to each node in this set to the product of the time function value and the scale factor. See command 17 for a description of the component options.

20. **INITial VELocity NODeset**, node set id, v_x, v_y, v_z

node set id	must match a node set on the GENESIS file
v_x, v_y, v_z	velocity vector

PRONTO initializes each component of the velocity of each node in this set to the specified value.

21. **INITial VELocity MATerial**, material id, v_x, v_y, v_z

material id	must match an element block on the GENESIS file
v_x, v_y, v_z	velocity vector

PRONTO initializes each component of the velocity of each node connected to the specified material block to the specified value.

22. **INITial VELocity ANGular**, material id, ω, c_x, c_y, c_z, n_x, n_y, n_z

material id	must match an element block on the GENESIS file
ω	angular velocity (radians per second)
c_x, c_y, c_z	center point coordinates
n_x, n_y, n_z	axis vector

PRONTO initializes the velocity of each node connected to the specified material block to correspond to the given angular velocity field. The velocity vector for each node is calculated by multiplying the cross-product of the normalized axis and the position vector from the center point to the node by the angular velocity.

23. **PRESsure**, side set id, function id, scale factor

side set id	must match a side set on the GENESIS file
function id	must match a function defined via command 15
scale factor	scales the function value (default = 1.0)

PRONTO applies a pressure each to the product of the time function value and the scale factor to each element side in this set. The calculated pressure value at each side node is multiplied by its side set scale factor as read from the GENESIS file. A positive pressure is directed inward to the element.

24. **MOVing PREssure**, side set id, c_x, c_y, c_z, peak id, rise id, C_p, t_0, scale factor

side set id	must match a side set on the GENESIS file
c_x, c_y, c_z	center point coordinates
peak id	must match a function defined via command 15
rise id	must match a function defined via command 15
C_p	propagation speed
t_0	arrival time
scale factor	scales the peak function value (default = 1.0)

PRONTO calculates the pressure at each node in the side set via Equation 7.15. A positive pressure is directed inward to the element.

25. **SILent BC**, side set id

side set id — must match a side set on the GENESIS file

A nonreflecting boundary condition is applied to each element side in this set.

26. **RIGid SURface**, side set id, c_x, c_y, c_z, n_x, n_y, n_z, μ

side set id — must match a side set on the GENESIS file
c_x, c_y, c_z — center point coordinates
n_x, n_y, n_z — outward normal vector
μ — static coefficient of friction
(default = 0.)

A rigid surface condition is enforced for all nodes in this side set.

27. **CONtact SURface**, side 1 id, side 2 id, μ_0, β, μ_1, γ

side 1 id — must match a side set on the GENESIS file
side 2 id — must match a side set on the GENESIS file
μ_0 — static coefficient of friction
(default = 0.)
β — kinematic partition factor
(default = 0.5)
μ_1 — high velocity coefficient of friction
(default = 0.)
γ — velocity decay coefficient

A contact condition is enforced between the two surfaces defined by the respective side sets. The kinematic partition is a relative weighting of the master slave relationship of the two surfaces. A value of zero implies that the first surface (defined by side 1 id) acts only as a master and the second surface acts only as a slave. A value of one reverses these roles. The default value (0.5) gives a symmetric treatment of the contact. If one surface is much more massive than the other, this variable should be adjusted so that it is treated as a master. By massive, we mean that the surface either has a higher material density and/or a coarser mesh refinement.

28. **MATerial**, material id, model, ρ

material id — must match an element block on the GENESIS file
model — valid material model name, the material models currently supported in PRONTO 3D are:

ELASTIC
ELASTIC PLASTIC
VISCOPLASTIC
DAMAGE

HYDRO
LOW DEN FOAM
SOIL N FOAMS
EP HYDRODYNAMIC

ρ material density

Appropriate material data for the given material model must be entered *immediately following* the MATERIAL command line. The data is entered in a keyword-value fashion: a *cue* followed by its assigned value. Each material type requires its own set of material cues. The material cues can be entered in any order, and on any number of input lines. An END statement is required to terminate the material data.

Listed below are the currently supported material models and their required material cues. Consult Chapter 4 for definitions of these parameters.

1.	ELASTIC	YOUNGS MODULUS POISSONS RATIO
2.	ELASTIC PLASTIC	YOUNGS MODULUS POISSONS RATIO YIELD STRESS HARDENING MODULUS BETA
3.	VISCOPLASTIC	YOUNGS MODULUS POISSONS RATIO YIELD STRESS HARDENING MODULUS GAMMA P
4.	DAMAGE	YOUNGS MODULUS POISSONS RATIO YIELD STRESS M K FRACTURE TOUGHNESS
5.	HYDRO	PRESSURE CUTOFF (positive in compression)
	A valid equation of state must be defined for this model via command 29	
6.	LOW DEN FOAM	YOUNGS MODULUS A B C

NAIR
P0
PHI

7. SOIL N FOAMS
BULK MODULUS
TWO MU
A0
A1
A2
PRESSURE CUTOFF (positive in compression)
FUNCTION ID; if this value is zero, the original bulk modulus is used. Otherwise, this must match a function defined via command 15 which defines pressure as a function of volume strain.

8. EP HYDRODYNAMIC
YOUNGS MODULUS
POISSONS RATIO
YIELD STRESS
HARDENING MODULUS
BETA
PRESSURE CUTOFF (positive in compression)

A valid equation of state must be defined for this model via command 29.

Examples of how the user might input the material data for the ELASTIC PLASTIC model are given below. They illustrate several different styles. All three examples yield identical results as far as PRONTO is concerned.

Example 1:

```
MATERIAL,1,ELASTIC PLASTIC,2.7E-3
HARDENING MODULUS = 30.E4
YOUNGS MODULUS = 30.E6
BETA = .5
POISSONS RATIO = .3
YIELD STRESS = 30.E3
END
```

Example 2:

```
MATERIAL,1,ELASTIC PLASTIC,2.7E-3
YOUNGS MODULUS = 30.E6 POISSONS RATIO = .3 BETA =.5
YIELD STRESS = 30.E3 HARDENING MODULUS = 30.E4
END
```

Example 3:

```
MATERIAL,1,ELASTIC PLASTIC,2.7E-3
YOUNGS MODULUS = 30.E6 POISSONS RATIO = .3 BETA = .5
YIELD STRESS = 30.E3 HARDENING MODULUS = 30.E4 END
```

29. **EQuation OF STate**, material id, eos

material id must match an element block on the GENESIS file
eos valid equation of state model name, the EOS models currently supported in PRONTO 3D are:

MG US-UP
MG POWER SERIES
JWL
IDEAL GAS

Appropriate material data for the given equation of state model must be entered *immediately following* the EQUATION OF STATE command line. The data is entered in a keyword-value fashion; a *cue* followed by its assigned value. Each model requires its own set of material cues. The material cues can be entered in any order, and on any number of input lines. An END statement is required to terminate the material data.

Listed below are the currently supported equation of state models and their required material cues. Consult Chapter 5 for definitions of these parameters.

1.	MG US-UP	C0
		S
		GAMMA
2.	MG POWER SERIES	K0
		K1
		K2
		GAMMA
3.	JWL	CD
		A
		B
		OMEGA
		R1
		R2
		ENERGY
4.	IDEAL GAS	GAMMA
		SOUND SPEED

An example material for the MG US-UP equation of state is given below. The MATERIAL command using the HYDRO material name is also shown. Note that the material id on the HYDRO command matches the material id on the EQUATION OF STATE command.

Example:

```
MATERIAL,8,HYDRO,2.7E-3
PRESSURE CUTOFF=-1.E9 $(note: pressure is negative in tension!)
END
EQUATION OF STATE,8,MG US-UP
CO=5380 S=1.337 GAMMA=2
END
```

30. **DETonation POint**, material no, c_x, c_y, c_z, t_0

    material id   material number of high explosive to be detonated; must match an element block on the GENESIS file

    c_x, c_y, c_z detonation point coordinates

    t_0 detonation time

31. **BURN CONstant**, bs

    bs   high explosive burn constant
    (default=2.5)

32. **DELete MATerial**, material id, deletion time

    material id   must match an element block on the GENESIS file

    deletion time   all elements in this material block are deactivated at this time

33. **DEATh**, material id, variable name, mode, level

    material id   must match an element block on the GENESIS file

    variable name   the critical variable may be one of the following:
    ENERGY
    VONMISES
    PRESSURE
    SIGMAX
    a state variable for this model from Table A.1

    mode   the criticality mode may be one of the following:
    MIN (minimum value)
    MAX (maximum value)
    ABS (absolute value)

    level   critical value

The adaptive element deletion capability requires a very mature user who understands how the selected material behaves. The capability built into PRONTO is quite general and allows elements to be deleted depending upon the level of energy, vonMises stress, pressure, maximum principal stress or any of the internal state variables for the material model. The criticality mode determines how failure occurs. The MIN mode signifies failure when the value of the critical variable falls below the critical level.

Failure occurs in the MAX mode when the value of the critical variable exceeds the critical level. ABS mode is similar to MAX mode, except that the absolute value of the critical variable is used.

For example, the command

```
DEATH = 3 , DAMAGE , MAX , .8
```

would delete elements within the material block with id 3 in which the damage exceeds a value of 0.8. Note that PRONTO would insist that this material block uses the DAMAGE model.

The user should be aware that it is possible to define nonsensical data by using a mode specification which is inappropriate for the critical variable. An example of this would be using the MIN specification with the VONMISES variable and a negative level.

Care must be taken to avoid deleting elements which have side boundary conditions applied to them!

B. STORAGE ALLOCATION FOR PRONTO 3D

B.1 Dimensioning Parameters and Variables

Name	Value	Description
MFIELD	22	maximum number of fields per line of input
NEBLK	64	maximum length of a vector block
NELNS	8	number of local nodes per element
NESNS	4	number of local nodes per element side
NHGM	4	number of hourglass shapes
NSPC	3	number of spatial coordinate components
NSYMM	6	number of components in a symmetric tensor

Name	Description
LCDATA	length of contact surface data heap
LCTEMP	length of contact surface scratch heap
MCONES	maximum number of equation of state constants
MCONS	maximum number of material constants
NACCBC	number of prescribed acceleration constraint conditions
NANGV	number of initial angular velocity specifications
NBCNOD	number of node sets
NBCSID	number of side sets
NCONT	number of contact surface pairs
NDEATH	number of adaptive element deletion specifications
NDETPT	number of detonation points
NEMBLK	number of material blocks
NFORCE	number of prescribed nodal point force load conditions
NFUNC	number of functions
NIVFLG	number of initial velocity by node set specifications
NIVMAT	number of initial velocity by material specifications
NMPBC	number of moving pressure load conditions
NNLIST	length of node set node list heap
NNOD	number of nodes
NODISP	number of no displacement constraint conditions
NPRBC	number of pressure load conditions
NQUIET	number of nonreflecting boundary conditions
NRIGID	number of rigid surfaces
NSLIST	length of side set element list heap

NSVLST	number of user selected state variables
NTOTFV	total number of user function data points
NTOTSV	length of internal state variables heap
NUMEL	number of elements
NVELBC	number of prescribed velocity constraint conditions

B.2 Nodal Point Arrays

In the following descriptions, the term *vector* identifies an array which stores the components of a Cartesian vector for each nodal point. The components are stored in the obvious order: X, Y, Z.

Name	Dimension	Description
COORD	(NNOD,NSPC)	Original Coordinate Vector
CUR	(NNOD,NSPC)	Current Coordinate Vector
DISPL	(NNOD,NSPC)	Displacement Vector
VEL	(NNOD,NSPC)	Velocity Vector
ACCL	(NNOD,NSPC)	Acceleration Vector
FORCE	(NNOD,NSPC)	Force Vector
XMASS	(NNOD)	Lumped Mass

B.3 Element Arrays

In the following descriptions, the term *tensor* identifies an array which stores the components of a Cartesian tensor for each element. For storage purposes, PRONTO categorizes tensors as general or symmetric. PRONTO stores the components of a *G-tensor* (general) as follows: XX, YX, ZX, XY, YY, ZY, XZ, YZ, ZZ. The components of an *S-tensor* (symmetric) are stored: XX, YY, ZZ, XY, YZ, ZX.

Name	Dimension	Description
LINK	(NELNS,NUMEL)	Connectivity List; global node number of each local node
SIG	(NSYMM,NUMEL)	Stress S-tensor
HGR	(NSPC,NHGM,NUMEL)	Hourglass Resistance Vectors; one vector for each hourglass shape per element
ELMASS	(NUMEL)	Mass
STRECH	(NSYMM,NUMEL)	Stretch S-tensor
ROTATE	(NSPC,NSPC,NUMEL)	Rotation G-tensor
RHO	(NUMEL)	Current Density
ENERGY	(NUMEL)	Internal Energy Density
VISPR	(NUMEL)	Bulk Viscosity Pressure

SV	(NTOTSV)	Internal State Variables Heap

Each material block is allocated a specific portion of the SV array whose structure depends upon the material model. The pointer IPSV locates this portion which is processed as SV(NINSV,NELB), where NINSV is the number of internal state variables per element for this material model and NELB is the number of elements in this material block. IPSV, NINSV, and NELB are defined for each material block within the KONMAT data structure.

Each material block with a material model which references an equation of state is also allocated a specific portion of the SV array for the storage of internal state variables for the equation of state. The structure of this array depends upon the equation of state. The pointer IPESV locates this portion which is processed as SVEOS(NESV,NELB), where NESV is the number of internal state variables per element for this equation of state and NELB is the number of elements in this material block. Note that if NESV is zero the pointer is not used. IPESV, NESV, and NELB are defined for each material block within the KONMAT data structure.

B.4 Optional Element Arrays

The following element arrays are only allocated if the user specifies certain options for which they are required, or the user specifies them on the plotting data base.

Name	Dimension	Description
STRAIN	(NSYMM,NUMEL)	Strain S-tensor; allocated if the strain flag KSFLG = 1
DOPT	(NSYMM,NUMEL)	Deformation Rate S-tensor; allocated if the strain rate flag KSRFLG = 1
STATUS	(NUMEL)	Activity Status; allocated if the status flag KSTAT = 1: 0 = inactive 1 = active

B.5 Material Block Arrays

All elements in PRONTO are processed in *Material Blocks*; contiguous groups of elements which share a common material model and data. The KONMAT data structure associates each material block with a material model.

The material model properties data structure PROP and equation of state properties data structure EOSDAT for each material block are stored within a column of the DATMAT array. This array is allocated rectangularly for convenience; the structure of each column depends upon its model. The material model definition interface returns the maximum number of material properties MCONS and equation of state properties MCONES that will be required. PRONTO adds 2 words to MCONS and 1 word to MCONES to store some reserved properties.

Each material block is assigned a deletion time. If the material block is to remain active, its deletion time is set beyond the termination time. A negative time indicates that a material block has been deactivated.

Name	Dimension	Description
KONMAT	(10,NEMBLK)	Material Block Data Structure: (1,N) = material id (2,N) = material type (3,N) = first element in block (4,N) = last element in block (5,N) = number of elements in block (6,N) = number of internal state variables (7,N) = IPSV, pointer into SV heap (8,N) = EOS type (if any) (9,N) = number of equation of state internal state variables (10,N) = IPESV, pointer into SV heap
DATMAT	(MCONS+MCONES,NEMBLK)	Material Properties Data Structure: (I, N) = PROP(I) ... (MCONS-1, N) = original dilatational modulus (MCONS, N) = original density (MCONS+I, N) = EOSDAT(I) ... (MCONS+MCONES,N) = original wave speed squared
DELETE	(NEMBLK)	Material Block Deletion Time

B.6 Contact Options

B.6.1 Contact Surface Arrays

Name	Dimension	Description
KLSURF	(20,NCONT)	Contact Surface Integer Data Structure (the following data occurs in pairs, one per surface): (1,N) = side set id (3,N) = pointer to side set element list (5,N) = pointer to side set node list (7,N) = pointer to data structures in KSLIST heap (9,N) = number of faces (11,N) = number of nodes (13,N) = maximum face connections (15,N) = constrained node list vector
CLSURF	(4,NCONT)	Contact Surface Real Data Structure: (1,N) = partition balance factor (2,N) = static coefficient of friction (3,N) = high velocity coefficient of friction (4,N) = decay factor
KSLIST	(LCDATA)	Contact Surface Data Heap; contains all data structures described in Section 6.2.2 for all contact surfaces.
CONDAT	(LCTEMP)	Contact Surface Scratch Heap; provides scratch storage for the contact algorithm.

B.6.2 Rigid Surface Arrays

Name	Dimension	Description
KRIGID	(3,NRIGID)	Rigid Surface Integer Data Structure: (1,N) = side set id (2,N) = pointer to node list (first in NSNODE, then in KSLIST) (3,N) = number of nodes
RIGID	(9,NRIGID)	Rigid Surface Real Data Structure: (1,N) = static coefficient of friction (2,N) = high velocity coefficient of friction

(3,N) = decay factor
(4,N) = surface point vector
(7,N) = surface outward unit normal vector

B.7 Kinematic Constraint Options

B.7.1 No Displacement Array

Name	Dimension	Description
KDISPL	(4,NODISP)	No Displacement Integer Data Structure: (1,N) = node set id (2,N) = pointer to node list (3,N) = component option: 1 = x 2 = y 3 = z (4,N) = number of nodes

B.7.2 Prescribed Velocity Arrays

Name	Dimension	Description
KPVELL	(5,NVELBC)	Prescribed Velocity Integer Data Structure: (1,N) = node set id (2,N) = pointer to node list (3,N) = function id, changed to function number in TELALL (4,N) = component option: 1 = x 2 = y 3 = z 4 = cylindrical 5 = tangential 6 = normal 7 = spherical (5,N) = number of nodes
PVBC	(8,NVELBC)	Prescribed Velocity Real Data Structure: (1,N) = scale factor (2,N) = origin vector (5,N) = velocity at last time step

(6,N) = unit normal vector

B.7.3 Prescribed Acceleration Arrays

Name	Dimension	Description
KPACCL	(5,NACCBC)	Prescribed Acceleration Integer Data Structure: (1,N) = node set id (2,N) = pointer to node list (3,N) = function id, changed to function number in TELALL (4,N) = component option: 1 = x 2 = y 3 = z (5,N) = number of nodes
PABC	(NACCBC)	Prescribed Acceleration Scale Factor

B.8 Load Options

B.8.1 Prescribed Nodal Force Arrays

Name	Dimension	Description
KFORCE	(5,NFORCE)	Prescribed Nodal Force Integer Data Structure: (1,N) = node set id (2,N) = pointer to node list (3,N) = function id, changed to function number in TELALL (4,N) = component option: 1 = x 2 = y 3 = z 4 = cylindrical 5 = tangential 6 = normal 7 = spherical (5,N) = number of nodes
PFORCE	(7,NFORCE)	Prescribed Nodal Force Real Data Structure: (1,N) = scale factor

(2,N) = origin vector
(5,N) = unit normal vector

B.8.2 Prescribed Pressure Arrays

Name	Dimension	Description
KPBC	(4,NPRBC)	Prescribed Pressure Integer Data Structure: (1,N) = side set id (2,N) = pointer to side set node list (3,N) = function id, changed to function number in TELALL (4,N) = number of nodes
PBCDAT	(NPRBC)	Prescribed Pressure Scale Factor

B.8.3 Moving Pressure Arrays

Name	Dimension	Description
KMPBC	(5,NMPBC)	Moving Pressure Integer Data Structure: (1,N) = side set id (2,N) = pointer to side set node list (3,N) = number of sides (4,N) = first function id, changed to function number in TELALL (4,N) = second function id, changed to function number in TELALL
PMPBC	(6,NMPBC)	Moving Pressure Real Data Structure: (1,N) = point vector (4,N) = scale factor (5,N) = arrival time (6,N) = propagation speed
XMPBC	(3,NSLIST)	Moving Pressure Nodal Data Structure (allocated for all side side nodes for efficiency): (1,N) = linear coefficient (2,N) = exponential coefficient (3,N) = delay time

B.9 Initial Velocity Options

B.9.1 Initial Node Set Velocity Arrays

Name	Dimension	Description
KVELFL	(3,NIVFLG)	Initial Velocity by Node Set Integer Data Structure: (1,N) = node set id (2,N) = pointer to node list (3,N) = number of nodes
VELFL	(NSPC,NIVFLG)	Initial Velocity by Node Set Vector

B.9.2 Initial Material Block Velocity Arrays

Name	Dimension	Description
KVELM	(NIVMAT)	Initial Velocity by Material Block ID
VELM	(NSPC,NIVFLG)	Initial Velocity by Material Block Vector

B.9.3 Initial Angular Velocity Arrays

Name	Dimension	Description
KANGV	(NANGV)	Initial Angular Velocity Material ID
ANGVEL	(6,NANGV)	Initial Angular Velocity Real Data Structure: (1,N) = angular velocity vector (4,N) = origin vector

B.10 Miscellaneous Options

B.10.1 User Function Arrays

PRONTO supports user supplied functions for a number of options. These functions are stored as an arbitrary number of abscissa-ordinate pairs. PRONTO user functions are generally treated as piecewise linear and monotonically increasing.

Name	Dimension	Description
KFDAT	(3,NFUNC)	User Function Integer Data Structure: (1,N) = function id (2,N) = number of data points

		(3,N) = pointer to the data points
FUNCS	(2,NTOTFV)	User Function Data Points

B.10.2 Detonation Point Arrays

Name	Dimension	Description
KDETPT	(NDETPT)	Detonation Point Material ID
DETPT	(4,NDETPT)	Detonation Point Real Data Structure: (1,N) = detonation point vector (4,N) = detonation time

B.10.3 Nonreflecting Boundary Array

Name	Dimension	Description
KQUIET	(4,NQUIET)	Nonreflecting Boundary Integer Data Structure: (1,N) = side set id (2,N) = pointer to side set element list (3,N) = pointer to side set node list (4,N) = number of sides

B.10.4 Adaptive Element Deletion Arrays

Name	Dimension	Description
KDEATH	(4,NDEATH)	Adaptive Element Deletion Integer Data Structure: (1,N) = material id (2,N) = material type (3,N) = deletion variable: -5 = internal energy density -6 = vonMises stress -7 = pressure -8 = maximum principal stress +I = internal state variable I (4,N) = mode of deletion: 1 = minimum 2 = maximum 3 = absolute value
DEATH	(NDEATH)	Adaptive Element Deletion Critical Value

B.11 Vector Block Arrays

To promote vectorization, PRONTO often processes in *vector blocks.* Large blocks of data, such as material blocks, are subdivided into vector blocks of length NEBLK or less. All elements in a vector block are processed simultaneously. A number of scratch arrays are used in these vector block processing loops. For this purpose, PRONTO allocates the Vector Block Scratch Array, SCREL(NEBLK,128). Various routines then access scratch storage by simply referencing a column of this array.

Note that the Vector Block Scratch Array and the Contact Surface Scratch Heap are coresident. PRONTO allocates memory for the larger of their respective dimensions.

B.12 Boundary Condition Sets

The following arrays are read directly from the GENESIS file, which is described in Reference [2].

B.12.1 Node Set Arrays

Name	Dimension	Description
KFLAGS	(NBCNOD)	Node Set ID
NPFLAG	(NBCNOD)	Node Set Node List Length
NFLOC	(NBCNOD)	Node Set Node List Pointer
IBC	(NNLIST)	Node Set Node List Heap; contains all the node set lists.
VALNOD	(NNLIST)	Node Set Node Factor Heap; contains a multiplication factor for each node set node.

B.12.2 Side Set Arrays

Name	Dimension	Description
NSFLG	(NBCSID)	Side Set ID
NSLEN	(NBCSID)	Side Set Element List Length
NVLEN	(NBCSID)	Side Set Node List Length
NSPTR	(NBCSID)	Side Set Element List Pointer
NVPTR	(NBCSID)	Side Set Node List Pointer
NELEMS	(NSLIST)	Side Set Node Element Heap; contains all the side set element lists.

NSNODE	(NESNS,NSLIST)	Side Set Node List Heap; contains all the side set node lists.
SVALUE	(NESNS,NSLIST)	Side Set Node Factor Heap; contains a multiplication factor for each side set node.

B.13 EXODUS Data Base

PRONTO allows the user to select what variables will be written to the EXODUS [3] output file. A default set of EXODUS output variables is defined, which may be overridden by user commands. The following arrays are allocated in the main routine to set up the data base.

B.13.1 Nodal Variable Output Arrays

Name	Dimension	Description
NODWR	(5)	Nodal Variable Option Flags, a 1 indicates the quantity is written: (1) = displacement (default=1) (2) = velocity (default=1) (3) = acceleration (default=1) (4) = lumped mass (default=0) (5) = reaction (default=0)
LISTND	(13)*8	Nodal Variable Option Names; contains the list of nodal variable names to be written on the EXODUS data base. The defaults are: (1) = DISPLX (2) = DISPLY (3) = DISPLZ (4) = VELX (5) = VELY (6) = VELZ (7) = ACCLX (8) = ACCLY (9) = ACCLZ (10) = null ... (13) = null

B.13.2 Element Output Arrays

Name	Dimension	Description
NELWR	(11)	Element Variable Option Flags, a 1 indicates the quantity is written: (1) = stress (default =1) (2) = cnergy (default=1) (3) = hourglass (default=0) (4) = strain (default=0) (5) = strech (default=0) (6) = rotation (default=0) (7) = ratedfm (default=0) (8) = density (default=0) (9) = pressure (default=0) (10) = vonmises (default=0) (11) = bulkq (default=0)
LISTEL	(25)	Element Variable Option Names; contains the list of element variable names to be written on the EXODUS data base. The defaults are: (1) = SIGXX (2) = SIGYY (3) = SIGZZ (4) = TAUXY (5) = TAUYZ (6) = TAUZX (7) = ENERGY (8) = null ... (25) = null

B.13.3 State Variable Output Arrays

Name	Dimension	Description
LISTSV	(MFIELD)	State Variable Output Names; contains the list of state variable names to be written on the EXODUS data base. There are no defaults.
MAPIE	(NEMBLK,NSVLST)	State Variable Option Map; a positive number indicates that a variable will be written.

Each internal state variable which the user specifies for the EXODUS data base is entered into the LISTSV array. Subroutine SVLIST is called for each user specified name, for each material block to search for a state variable match. If a match is found, the internal index of that variable is entered into the MAPIE array. If a particular material model does not have that internal state variable. a zero is entered into the map. This mapping is required because totally different material models may have the same internal state variable (e.g., equivalent plastic strain), but in different locations.

C. ADDING A NEW CONSTITUTIVE MODEL TO PRONTO

PRONTO was designed from the beginning to serve as a testbed for new constitutive models and algorithms. We have incorporated a material .interface subroutine which allows you (the constitutive model developer) to add a new material model with very little effort. We have purposely designed this interface so that you do not have to understand the inner workings of the finite element code, especially with respect to the allocation and management of computer memory. If the instructions in subroutine MATINT are followed correctly, the computer program will handle all memory allocation, material data reading, and material data printing. There are three steps that you should follow to add a new model.

STEP I

Subroutine MATINT contains instructions via comment cards within the FORTRAN which outline the steps the you should follow to add your new material model. Most of the changes required involve adding or changing numbers in DATA or PARAMETER statements. Since we have no foreknowledge of what the material constants represent for a particular material, we require that a few lines of FORTRAN be added which tell the code what the initial dilatational modulus ($\lambda + 2\mu$) is for the material. This value must be stored in the variable DATMOD in step 12. At the same place in the code, you can calculate any combinations of the input material constants that may be required in the constitutive subroutine. (e.g. bulk modulus from Young's modulus and Poisson's ratio).

You are restricted to twenty characters for your material name, material cues and internal state variable names which you define in subroutine MATINT. The names may have have multiple words, separated by blanks (i.e., YOUNGS MODULUS). You must define names such that each word is unique to the first three characters. This means that you may define material cues C1, C2, C3, etc. but not CON1, CON2, CON3, etc. All words must be uppercase only since the free field reader is case insensitive. Finally, please do not use special characters in your words as we cannot guarantee the results.

We reiterate that all of the steps which you must take when adding a new material model are outlined in detail via comments within the FORTRAN in subroutine MATINT.

STEP II

This step is optional; it is only required if the new material model contains internal state variables which must be initialized to some value other than zero (we initialize all internal state variables to zero by default). If state variables must be initialized, you must add an ELSE IF block to subroutine SVINIT for this material. This block should read:

```
ELSE IF( MKIND .EQ. # ) THEN
                 .
         initialize your internal
         state variables here
                 .
                 .
                 .
```

Your material number, #, corresponds to the position where your material resides within the list of materials defined in subroutine MATINT. Generally, a new material is added at the end of the list; your material number would then be the same as the number of defined materials in MATINT, which you must increment in step I. This step should be obvious from looking at how other material models are coded in SVINIT. Please use comments for your material.

STEP III

You must add in subroutine UPDSTR the call to your material subroutine. You may call your material subroutine any name you wish, but we recommend our convention: MAT#, where # is your material number as described above under STEP II. The call is included by adding an ELSE IF block to subroutine UPDSTR which should read:

```
ELSE IF( MKIND .EQ. # ) THEN
   CALL your subroutine( .... your argument list .... )
```

This step should be obvious from looking at how other material models are coded in UPDSTR. Please use comments for your material so that years from now we have some chance of figuring out what was added to the code.

References

[1] Taylor, L. M. and Flanagan, D. P., "PRONTO 2D A Two Dimensional Transient Solid Dynamics Program," SAND86–0594, Sandia National Laboratories, Albuquerque, NM, March 1987.

[2] Taylor, L. M., Flanagan, D. P., and Mills-Curran, W. C., "The GENESIS Finite Element Mesh File Format," SAND86–0910, Sandia National Laboratories, Albuquerque, NM, May 1986.

[3] Mills-Curran, W. C., "EXODUS: A Finite Element Fiie Format for Pre- and Post-processing," SAND87–2977, Sandia National Laboratories, Albuquerque, NM, September 1988.

[4] *American National Standard Programming Language FORTRAN* ANSI X3.9–1978, American National Standards Institute, New York, 1978.

[5] Flanagan, D. P., Mills-Curran, W. C., and Taylor, L. M., "SUPES - A Software Utilities Package for the Engineering Sciences," SAND86–0911, Sandia National Laboratories, Albuquerque, NM, September 1986.

[6] Dienes, J. K., "On the Analysis of Rotation and Stress Rate in Deforming Bodies," *Acta Mechanica*, vol. 32, pp. 217–232, 1979.

[7] Johnson, G. C. and Bammann, D. J., "On the Analysis of Rotation and Stress Rate in Deforming Bodies," *International Journal of Solids and Structures*, vol. 20, no. 8, pp. 725–737, 1984.

[8] Flanagan. D. P. and Taylor, L. M., "On the Analysis of Rotation and Stress Rate in Deforming Bodies," *Computer Methods in Applied Mechanics and Engineering*, vol. 62, pp. 305–320, 1987.

[9] Truesdell, C., *The Elements of Continuum Mechanics.* New York: Springer Verlag, 1966.

[10] Flanagan, D. P. and Belytschko, T., "A Uniform Strain Hexahedron and Quadrilateral with Orthogonal Hourglass Control," *International Journal for Numerical Methods in Engineering*, vol. 17, pp. 679–706, 1981.

[11] Bathe, K. J. and Wilson, E. L., *Numerical Methods in Finite Element Analysis.* New Jersey: Prentice-Hall, 1976.

[12] Hughes, T. J. R. and Winget, J., "Finite Rotation Effects in Numerical Integration of Rate Constitutive Equations Arising in Large-Deformation Analysis," *International Journal for Numerical Methods in Engineering*, vol. 15, no. 12, pp. 1862–1867, 1980.

[13] Flanagan, D. P. and Belytschko, T., "Eigenvalues and Stable Time Steps for the Uniform Strain Hexahedron and Quadrilateral," *ASME Journal of Applied Mechanics*, vol. 51, no. 1, pp. 35–40, 1984.

[14] Krieg, R. D. and Krieg, D. B., "Accuracies of Numerical Solution Methods for the Elastic-Perfectly Plastic Model," *ASME Journal of Pressure Vessel Technology*, vol. 99, pp. 510–515, 1977.

[15] Schreyer, H. L., Kulak, R. F., and Kramer, J. M., "Accurate Numerical Solutions for Elastic-Plastic Models," *ASME Journal of Pressure Vessel Technology*, vol. 101, pp. 226–234, 1979.

[16] Taylor, L. M. and Becker, E. B., "Some Computational Aspects of Large Deformation, Rate-Dependent Plasticity Problems," *Computer Methods in Applied Mechanics and Engineering*, vol. 41, no. 3, pp. 251–277, 1983.

[17] Perzyna, P., "Fundamental Problems in Viscoplasticity," in *Recent Advances in Applied Mechanics*, pp. 243–377, New York: Academic Press, 1966.

[18] Kipp, M. E. and Grady, D. E., "Numerical Studies of Rock Fragmentation," SAND79-1582, Sandia National Laboratories, Albuquerque, NM, 1978.

[19] Taylor, L. M., Chen, E. P., and Kuszmaul, J. S., "Microcrack-Induced Damage Accumulation in Brittle Rock Under Dynamic Loading," *Computer Methods in Applied Mechanics and Engineering*, vol. 55, no. 3, pp. 301–320, 1986.

[20] Kuszmaul, J. S., "A New Constitutive Model for Fragmentation of Rock Under Dynamic Loading," in *Proceedings of the Second International Symposium on Fragmentation by Blasting*, (Keystone, CO), pp. 412–423, August 1987.

[21] Budiansky, B. and O'Connell, R. J., "Elastic Moduli of a Cracked Solid," *Computer Methods in Applied Mechanics and Engineering*, vol. 12, pp. 81–97, 1976.

[22] Grady, D., "The Mechanics of Fracture Under High-rate Stress Loading," in *William Prager Symposium on Mechanics of Geomaterials: Rocks, Concretes and Soils*, (Bazant, Z. P., ed.), 1983.

[23] Kuszmaul, J. S., "A Technique for Predicting Fragmentation and Fragment Sizes Resulting from Rock Blasting," in *Proceedings of the 28th U.S. Symposium on Rock Mechanics*, (Tucson, AZ), 1987.

[24] Kipp, M. E., Grady, D. E., and Chen, E. P., "Strain-rate Dependent Fracture Initiation," *International Journal of Fracture*, vol. 16, pp. 471–478, October 1980.

[25] Krieg, R. D., "A Simple Constitutive Description for Soils and Crushable Foams," SC-DR-72-0883, Sandia National Laboratories, Albuquerque, NM, 1978.

[26] Swenson, D. V. and Taylor, L. M., "A Finite Element Model for the Analysis of Tailored Pulse Stimulation of Boreholes," *International Journal for Numerical and Analytical Methods in Geomechanics*, vol. 7, pp. 469–484, 1983.

[27] Neilsen, M. K., Morgan, H. S., and Krieg, R. D., "A Phenomenological Constitutive Model for Low Density Polyurethane Foams," SAND86-2927, Sandia National Laboratories, Albuquerque, NM, April 1987.

[28] Kipp, M. E. and Lawrence, R. J., "WONDY V - A One-Dimensional Finite Difference Wave Propagation Code," SAND81-0930, Sandia National Laboratories, Albuquerque, NM, June 1982.

[29] Swegle, J. W., "TOODY-IV - A Computer Program for Two-Dimensional Wave Propagation," SAND78-0552, Sandia National Laboratories, Albuquerque, NM, September 1978.

[30] Lee, E., Finger, M., and Collins, W., "JWL Equation of State Coefficients for High Explosives," UCID-16189, Lawrence Livermore National Laboratory, Livermore, CA, January 1973.

[31] Dobratz, B. M., "LLNL Explosives Handbook, Properties of Chemical Explosives and Explosive Simulants," UCRL-52997, Lawrence Livermore National Laboratory, Livermore, CA, March 1981.

[32] Key, S. W., Beisinger, Z. E., and Krieg, R. D., "HONDO II - A Finite Element Computer Program for the Large Deformation Dynamic Response of Axisymmetric Solids," SAND78-0422, Sandia National Laboratories, Albuquerque, NM, October 1978.

[33] Lysmer, J. and Kuhlemeyer, R. L., "Finite Dynamic Model for Infinite Media," *Journal of the Engineering Mechanics Division of ASCE*, pp. 859-877, August 1979.

[34] Cohen, M. and Jennings, P. C., "Silent Boundary Methods," in *Computational Methods for Transient Analysis*, (Belytschko, T. and Hughes, T. J. R., eds.), North-Holland, 1983.

Distribution:

Dr. R. T. Allen
Pacifica Technology
P.O. Box 148
Del Mar, CA 92014

Prof. S. Atluri
Center for the Advancement of
Computational Mechanics
School of Civil Engineering
Georgia Institute of Technology
Atlanta, GA 30332

Dr. William E. Bachrach
Areojet Research Propulsion Inst.
P. O. Box 13502
Sacramento, CA 95853-4502

Prof. E. B. Becker (3)
Prof. J. T. Oden
Prof. M. Stern
Department of Aerospace Engineering
and Engineering Mechanics
University of Texas
Austin, TX 78712

Prof. T. Belytschko
Department of Civil Engineering
Northwestern University
Evanston, IL 60201

Mr. Chuck Charman
GA Technologies
P.O. Box 81608
San Diego, CA 92138

Mr. Ken K. Chipley
Martin Marietta Energy Systems
P.O. Box 2009
Oak Ridge, TN 37831-8053

Mr. Ken P. Chong
Dept. of Civil Engineering
University of Wyoming
Laramie, WY 82071

Dr. S. C. (Tony) Chou
U.S. Army Materials Technology Lab
SLCMT-BM
Watertown, MA 02172-0001

Mr. Dwight Clark
Morton Thiokol Corp.
P. O. Box 524
Mail Stop 281
Brigham City, UT 84302

Mr. Gerald Collingwood
Morton Thiokol
Huntsville, AL 35807

C. H. Conley
School of Civil and
Environmental Engineering
Hollister Hall
Cornell University
Ithaca, NY 14853

Mr. Bill Cook
Los Alamos National Laboratory
Los Alamos, NM 87545

Dr. Ian Cullis
XTZ Division
Royal Armament R&D Establishment
Fort Halstead
Sevenoaks, Kent, United Kingdom

Richard E. Danell
Research Officer
Central Research Laboratories
BHP Research & New Technology
P.O. Box 188
Wallsend NSW 2287, Australia

Mr. Ramji Digumarthi
Org. 8111, Bldg. 157
Lockheed MSD
P.O. Box 3504
Sunnyvale, CA 94088-3504

Mr. J. Donald Dixon
Spokane Research Center
U.S. Bureau of Mines
315 Montgomery Avenue
Spokane, WA 99207

Dr. R. S. Dunham (2)
Dr. Joe Rashid
Anatech International Corp.
3344 N. Torrey Pines Ct.
Suite 320
La Jolla, CA 92037

Dr. Russel Garnsworthy
CRA Advanced Tech Development
G.P.O. Box 384D
Melbourne 3001, Australia

Dr. Gerry Goudreau
Methods Development Group
Mechanical Engineering Department
Lawrence Livermore National Lab
Livermore, CA 94550

Dr. Francisco Guerra
Mail Stop C931
WX11 Division
Los Alamos National Laboratory
Los Alamos, NM 87545

Dr. John Hallquist
Shock Physics Group, L-200
Lawrence Livermore National Lab
Livermore, CA 94550

Dr. David Hibbitt (2)
Dr. Joop Nagtegaal
Hibbitt, Karlsson & Sorrensen, Inc.
100 Medway St.
Providence, RI 02906

Mr. Jeffery P. Hill (2)
Mail Stop F644
Group CTR-4
Los Alamos National Laboratory
Los Alamos, NM 87545

Mr. Roger Hill
Mail Stop D449
P15 Division
Los Alamos National Laboratory
Los Alamos, NM 87545

Mr. John Hopson
Group T3
Mail Stop B216
Los Alamos National Laboratory
Los Alamos, NM 87545

Dr. William Hufferd
United Technologies
Chemical Systems Division
P.O. Box 50015
San Jose, CA 95150-0015

Prof. T. J. R. Hughes
Department of Mechanical Engineering
Stanford University
Palo Alto, CA 94306

James P. Johnson
Technology Development
CPC Analysis Dept.
General Motors Corp.
Engineering Center
30003 Van Dyke Avenue
Warren, MI 48090-9060

Dr. Rembert Jones and (2)
Dr. Alan S. Kushner
Office of Naval Research
Structural Mechanics Div. (Code 434)
800 N. Quincy Street
Arlington, VA 22217

Dr. Gordon R. Johnson
Honeywell Inc.
5901 S. County Rd. 18
Edina, MN 55436

Mr. James Johnson
Rm L120, CPC Engineering Center
30003 Van Dyke
Warren, MI 48090

Dr. Mike Katona
TRW Ballistic Missiles Division
P.O. Box 1310
Bldg 527, Rm 709
San Bernadino, CA 92402

Mr. Gary Ketner
Research Engineer
Applied Mechanics and Structures
Battelle Pacific Northwest Laboratories
P.O. Box 999
Richland, WA 99352

Dr. Sam Key
McNeal-Schwendler Corp.
815 Colorado Blvd.
Los Angeles, CA 90041

Prof. J. K. Lee
Department of Engineering Mechanics
Ohio State University
Columbus, OH 43210

Mr. Richard Lung
TRW Corporation
P. O. Box 1310
Bldg 527, Rm 709
San Bernadino, CA 91763

Mr. Craig Miller
Unit 973
General Electric Company
Neutron Devices Department
P.O. Box 2908
Largo, FL 34294-2908

Mr. J. J. Murphy (5)
Vehicle Technology 59-22
Bldg 580
Lockheed Missiles and Space Co.
P. O. Box 3504
Sunnyvale, CA 94088

Prof. V. D. Murty
5000 N. Willamette Blvd.
School of Engineering
University of Portland
Portland, OR 97203

Prof. S. Nemat-Nasser (2)
Prof. Dave Benson
Department of Applied Mechanics
and Engineering Sciences
University of California
San Diego
La Jolla, CA 92093

Mr. Jim Nemes
Code 6331
Naval Research Lab
Washington, DC 20375-5000

Dr. R. E. Nickell
c/o Anatech International Corp.
3344 N. Torrey Pines Court
Suite 320
La Jolla, CA 92037

Mr. Dean Norman
Waterways Experiment Station
P.O. Box 631
Vicksburg, MS 39180

Dr. Robert Pardue
Martin Marietta
Y-12 Plant, Bldg. 9998
Mail Stop 2
Oak Ridge, TN 37831

Mr. Mitchell R. Phillabaum
Monsanto Research Corp.
MRC-MOUND
Miamisburg, OH 45342

Dr. Harold E. Read
S-Cubed
P.O. Box 1620
LaJolla, CA 92038-1620

Mr. B. Rebourcet
MA/MCN
Commissariat a L'Energie Atomique
Centre D'Etudes de Limeil
Boite Postale 27
94190 Villeneuve St Georges
France

Mr. Samit Roy
Research Engineer
Dept. of Engineering Mechanics
Southwest Research Institute
P.O. Drawer 28510
San Antonio, TX 78284

Mr. R. Ruppin
Israel Atomic Energy Commission
Soreq Nuclear Research Centre
70600 Yavne, Israel

Mr. Donald W. Sandidge
U.S. Army Missile Command
AMSMI-RLA
Redstone Arsenal, AZ 35898-5247

Mr. Steve Sauer
K-Tech Corporation
901 Pennsylvania, N.E.
Albuquerque, NM 87110

Mr. Ray Stoudt
Lawrence Livermore National Lab
P.O. Box 808, L200
Livermore, CA 94550

Prof. D. V. Swenson
Mechanical Engineering Dept.
Durland Hall
Kansas State University
Manhattan, KS 66506

Mr. Sing C. Tang
P. O. Box 2053
Rm 3039 Scientific Lab
Dearborn, MI 48121-2053

Mr. J. Terrier
Societe Nationale
Des Poudres et Explosifs
Centre de Recherches du Bouchet
91710 Vert le Petit
France

Mr. David Wade, 36E
Bettis Atomic Power Laboratory
P.O. Box 79
West Miffland, PA 15122

Dr. Paul T. Wang
Fabricating Technology Division
Aluminum Company of America
Alcoa Technical Center
Alcoa Center, PA 15069

Prof. Tomasz Wierzbicki
Dept. of Ocean Engineering
Bldg. #5-218
M.I.T.
Cambridge, MA 02139

Mr. J. A. Zukas
Computational Mechanics
Consultants, Inc.
8600 La Salle Road
Suite 614
Towson, MD 21204

1510	J. W. Nunziato
1511	D. K. Gartling
1520	L. W. Davison
1521	R. D. Krieg (12)
1522	R. C. Reuter (15)
1523	J. H. Biffle (40)
1523	D. P. Flanagan (10)
1523	L. M. Taylor (10)
1524	A. K. Miller (12)
1530	D. B. Hayes
1531	J. W. Swegle
1531	S. L. Thompson
1533	M. E. Kipp
1533	S. T. Montgomery
1533	A. C. Robinson
1533	P. Yarrington
1534	J. R. Asay
1550	C. W. Peterson
2513	S. F. Roller
2541	Alex Gonzales
3141	S. A. Landenberger (5)
3151	W. I. Klein (3)
3154-1	For DOE/OSTI (8)
5165	N. R. Hansen
6253	N. R. Warpinski
6258	P. J. Hommert
6258	J. S. Kuszmaul
6258	B. J. Thorne
6313	B. A. Luke
6322	R. E. Glass
6322	M. A. Kincy
6323	R. H. Yoshimura
7111	J. S. Phillips
7112	A. J. Chabai
7133	H. C. Walling
7541	R. G. Bell
8240	C. W. Robinson
8241	M. L. Chiesa
8241	K. J. Perano (5)
8242	V. C. Prantil
8243	D. J. Bammann
8243	L. A. Bertram
8243	M. L. Callabresi
8243	W. E. Mason, Jr.
8524	J. A. Wackerly
9113	J. T. Schamaun
9122	M. J. Forrestal
9141	M. H. Gubbels

☆ U.S. GOVERNMENT PRINTING OFFICE: 1989—673-049/81031